МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
СЄВЄРОДОНЕЦЬКИЙ ТЕХНОЛОГІЧНИЙ ІНСТИТУТ

КОНСПЕКТ ЛЕКЦІЙ

по курсу

"КОМП'ЮТЕРНАЯ ЛОГІКА "
(Частина 2)

напрям підготовки 6.050102 "Комп'ютерна інженерія».

"До друку дозволяю"
Зам. директора
по учбово-
методичній та О.М. Брежнєв
виховній роботі _____

Протокол № від 2013

Розробни к В.О. Лифар

Весь цифровий і фактичний матеріал,

бібліографічні відомості перевірені.

Написання одиниць відповідає стандартам

Сєвєродонецьк СНУ СТІ 2014

УДК 681.324

Конспект лекцій по дисципліні “Компьютерная логика“ “ (для студентів, що навчаються за напрямком підготовки 6.050102 “Комп’ютерна інженерія ”Сост.: В. А. Лыфарь 83 с.

Розробник: В.О. Лифар, доцент, к.т.н.

Відп. за видання

Рецензент А. І. Рязанцев, професор, д.т.н.

СОДЕРЖАНИЕ

1 АБСТРАКТНЫЕ ЦИФРОВЫЕ АВТОМАТЫ

1.1 Основные понятия

Цифровой автомат можно трактовать как устройство, осуществляющее прием, хранение и преобразование дискретной информации по некоторому алгоритму. С определенной точки зрения к автоматам можно отнести как реальные устройства (ЭВМ), так и абстрактные системы (математические модели).

Автоматом называется дискретный преобразователь информации, способный принимать различные состояния, переходить под воздействием входных сигналов из одного состояния в другое и выдавать выходные сигналы.

Общая теория автоматов подразделяется на абстрактную и структурную.

Абстрактная теория автоматов, отвлекаясь от структуры автомата (т.е. не интересуясь способом его построения), изучает лишь поведение автомата относительно внешней среды. Абстрактная теория автоматов близка к теории алгоритмов, являясь по существу ее дальнейшей реализацией.

В противоположность абстрактной теории автоматов, структурная теория автоматов интересуется как структурой самого автомата, так и структурой входных воздействий и реакций автомата на них. В структурной теории изучаются способы построения автоматов, способы кодирования входных воздействий и реакций автомата на них. Структурная теория автоматов является продолжением и развитием абстрактной теории. Опираясь на аппарат булевых функций и на абстрактную теорию автоматов, структурная теория дает эффективные рекомендации по разработке реальных устройств вычислительной техники.

Абстрактный цифровой автомат (ЦА) является математической моделью дискретного управляющего устройства.

Абстрактный ЦА определяется множеством, состоящим из шести элементов:

$S=\{ X,A,Y,\delta,\lambda,a_o\}$, где:

$X=\{x_1, x_2,... x_n\}$ - множество входных сигналов (входной алфавит);

$Y=\{y_1, y_2, ... y_m\}$ - множество выходных сигналов (выходной алфавит);

$A=\{ a_0,a_1, a_2,... a_N\}$ - множество состояний (алфавит состояний);

a_o- начальное состояние ($a_o \in A$);

δ - функция переходов автомата, задающая отображение $(X \times A) \to A$, т.е. ставящая в соответствие любой паре элементов декартового произведения $(X \times A)$ элемент множества A;

λ - функция выходов автомата, задающая либо отображение $(X \times A) \to Y$, либо отображение $A \to Y$.

Иными словами, функция переходов δ показывает, что автомат S, находясь в некотором состоянии $a_j \in A$, при появлении входного сигнала $x_j \in X$ переходит в некоторое состояние $a_p \in A$. Это можно записать:

$a_p=\delta(a_i,X_j)$.

Функция выходов λ показывает, что автомат S, находясь в некотором состоянии $a_j \in A$, при появлении входного сигнала $x_j \in X$ выдает выходной сигнал $y_k \in Y$. Это можно записать: $y_k = \lambda(a_i, X_j)$.

Понятие **состояние** в определение автомата было введено в связи с тем, что часто возникает необходимость в описании поведения систем, выходы которых зависят не только от состояния входов в данный момент времени, но и от некоторой предыстории, т.е. от сигналов, которые поступали на входы системы ранее. Состояние как раз и соответствует некоторой памяти о прошлом, позволяя устранить время как явную переменную и выразить выходные сигналы как функцию состояний и входов в данный момент времени.

Абстрактный автомат функционирует в дискретном автоматном времени t=0,1,2,... и переходы из состояния в состояние осуществляются мгновенно. В каждый момент t дискретного времени автомат находится в определенном состоянии a(t) из множества A состояний автомата, причем в начальный момент времени t=0 он всегда находится в начальном состоянии a_o. В момент времени t, будучи в состоянии a(t), автомат способен воспринять на входном канале сигнал x(t) ∈ X, и выдать на выходном канале сигнал y(t)= λ (a(t), x(t)), переходя в состояние a(t+1)= = δ (a(t),x(t)). Зависимость выходного сигнала от входного и от состояния означает, что в его составе имеется память.

1.2 Типы абстрактных автоматов

По способу формирования функции выходов выделяют три типа абстрактных автоматов: автомат Мили, автомат Мура, С-автомат. Автомат Мили характеризуется системой уравнений:

y(t)= λ (a(t), x(t));
a(t+1)= δ(a(t), x(t)). (1-1)

Автомат Мура - системой уравнений:

y(t)= λ (a(t));
a(t+1)= δ(a(t), x(t)). (1-2)

С-автомат - системой уравнений:

y= $y_1 \cup y_2$
$y_1(t)= \lambda_1 (a(t),x(t))$;
$У_2(t)= \lambda_2(a(t))$;
a(t+1)=δ(a(t),x(t)).

Произвольный абстрактный автомат Мили или Мура (рис. 1.1.) имеет один входной и один выходной каналы. Произвольный абстрактный С-автомат имеет один входной и два выходны х канала (рис. 1.2.).

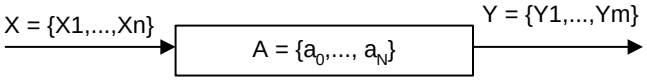


Рисунок 1.1- Абстрактный автомат

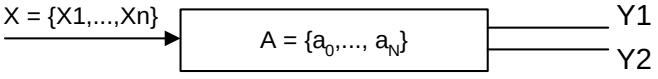


Рисунок. 1.2 Абстрактный С-автомат

Если на вход абстрактного автомата Мили или Мура, установленного в начальное состояние a_o, подавать буква за буквой некоторую последовательность букв входного алфавита x(0), x(1), ...- входное слово, то на выходе автомата будут последовательно появляться буквы выходного алфавита y(0), y(1),...- выходное слово. Для случая С-автомата на его выходах будут появляться две последовательности: $y_1(0)$, $y_1(1)$,... и $y_2(0)$, $y_2(1)$,.... В абстрактном С- автомате выходной сигнал $y_2(t)= \lambda_2 (a(t))$ выдается все время, пока автомат находится в состоянии a(t). Выходной сигнал $y_1(t)=\lambda_1(a(t),x(t))$ выдается во время действия входного сигнала x(t) при нахождении С-автомата в состоянии a(t).

Таким образом, на уровне абстрактной теории функционирование цифрового автомата понимается как преобразование входных слов в выходные слова.

Выделяют полностью определенные и частичные автоматы.

Полностью определенным называется абстрактный цифровой автомат, у которого функция переходов или функция выходов, или обе эти функции определены для всех пар переходов (x_i,a_j).

Частичным называется абстрактный цифровой автомат, у которого функция переходов или функция выходов, или обе эти функции определены не для всех пар переходов (x_i,a_j).

Абстрактный цифровой автомат называется инициальным, если на множестве его состояний A выделяется начальное состояние a_o.

1.3 Способы задания абстрактных автоматов

Чтобы задать конечный автомат S, необходимо описать все элементы множества: S={ X,A,Y, δ , λ ,a_o}. Существует несколько способов задания работы автомата, но наиболее часто используется табличный (матричный), графический, аналитический.

При табличном способе автомат задается двумя таблицами: таблицей переходов и таблицей выходов, или матрицей соединений. Таблица переходов произвольного полностью определенного автомата строится следующим образом: строки таблицы помечаются буквами входного алфавита автомата, а столбцы таблицы - буквами алфавита состояний автомата; В клетке таблицы переходов, находящейся на пересечении строки, отмеченной входным сигналом x_i, и столбца отмеченного состоянием a_j, ставится состояние a_k, являющееся результатом перехода автомата из состояния a_j под воздействием входного сигнала x_i, что определяется выражением $a_k = \delta(a_j, x_j)$.

Таблица 1.1. Таблица переходов автомата

Состояния / Входные сигналы	a_1	a_2		a_k
x_1	$\delta(a_1,x_1)$	$\delta(a_2, x_1)$	...	$\delta(a_к, x_1)$
…				
x_j	$\delta(a_1,x_j)$	$\delta(a_2, x_j)$		$\delta(a_к, x_j)$

Пример заполнения таблицы переходов некоторого абстрактного полностью определенного автомата с входным алфавитом X={x_1, x_2}- и алфавитом состояний A={a_1, a_2, a_3} представлен в табл. 1.2.

Таблица 1.2

A / x	a_1	a_2	a_3
x_1	a_2	a_3	a_1
x_2	a_1	a_1	a_2

Если абстрактный автомат частичный, то в клетке таблицы его переходов, находящейся, на пересечении строки, отмеченной входным сигналом и столбца отмеченного соответствующим состоянием (при условии, что переход в это состояние под действием данного входного сигнала не определен) ставится прочерк, и любое входное слово, приводящее к указанному переходу является запрещенным.

Заполнение остальных клеток аналогично случаю полностью определенного автомата. Вид таблицы переходов не зависит от типа задаваемого автомата (автомат Мили, Мура, С-автомат). Таблицы выходов автоматов Мили, Мура, С-автомата имеют различия.

Таблица выходов полностью определенного автомата Мили строится следующим образом: идентификация столбцов и строк, а также формат таблицы соответствуют таблице переходов полностью определенного автомата. В клетке таблицы выходов, находящейся на

пересечении строки, отмеченной входным сигналом X_j, и столбца, отмеченного состоянием $a_к$, ставится выходной сигнал $У_m$, который автомат выдает, находясь в состоянии a_k при наличии входного сигнала XJ, что определяется выражением $Ym=\lambda(a_k, x_j)$.

Таблица 1.3. Таблица выходов абстрактного автомата Мили.

Состояния / Входные сигналы	a_1	a_2		a_k
x_1	$\lambda(a_1,x_1)$	$\lambda(a_2, x_1)$		$\lambda(a_k, x_1)$
…		….	...	
x_i	$\lambda(a_1,x_i)$	$\lambda(a_2, x_i)$		$\lambda(a_к, x_i)$

Пример заполнения таблицы выходов некоторого абстрактного полностью определенного автомата с входным алфавитом $X=\{x_1, x_2\}$,алфавитом состояний $A=\{a_1, a_2,a_3\}$ и выходным алфавитом $Y=\{y_1, y_2, y_3\}$-представлен в табл. 1.4.

Таблица 1.4

a / x	a_1	a_2	a_3
x_1	y_2	y_3	y_1
x_2	y_3	y_1	y_2

Таблица выходов полностью определенного автомата Мура строится более просто: каждому состоянию автомата ставится в соответствие свой выходной сигнал. Пример таблицы выходов автомата Мура с алфавитом состояний $A=\{a_1, a_2, a_3\}$ и выходным алфавитом $Y=\{y_1, y_2, y_3\}$ представлен в таблице 1.5.

Таблица 1.5.

A	a_1	a_2	a_3
y	y_1	y_2	y_2

Очевидно, абстрактный полностью определенный С-автомат задается двумя таблицами выходов, первая из которых есть таблица выходов автомата Мили, а вторая - Мура. Если автомат частичный, то в некоторых клетках его таблицы может стоять прочерк, что означает отсутствие выходного сигнала.

На практике таблицы переходов и выходов часто совмещают в одну таблицу, называемую отмеченной таблицей переходов автомата. Примеры отмеченных таблиц переходов представлены в табл. 1.6. -1.8. (Общий вид отмеченной таблицы переходов - табл. 1.6., отмеченная таблица переходов автомата Мили - табл. 1.7., отмеченная таблица переходов автомата Мура - табл. 1.8.).

Кроме рассмотренных выше таблиц переходов и выходов произвольный абстрактный автомат может быть задан матрицей соединений.

Матрица соединений является квадратной и содержит столько столбцов (строк), сколько различных состояний содержит алфавит состояний данного автомата. Каждый столбец (строка) матрицы соединений помечается буквой состояния автомата. В клетке, находящейся на пересечении столбца, помеченного буквой a_j и строки, помеченной буквой a_s автомата, ставится входной сигнал (или дизъюнкция входных сигналов), под воздействием которого осуществляется данный переход.

Для абстрактного автомата Мили в клетке рядом с состоянием проставляется также выходной сигнал, который автомат выдает в результате данного перехода (табл. 1.9.) Для автомата Мура выходной сигнал проставляется в строке рядом с состоянием (эти состояния соответствуют исходным состояниям автомата).

Таблица 1.6

Состояния / Входные сигналы	a_1	a_2	…	a_k
x_1	$\delta(a_1,x_1)$	$\delta(a_2,x_1)$	…	$\delta(a_k,x_1)$
	$\lambda(a_1,x_1)$	$\lambda(a_2,x_1)$	…	$\lambda(a_k,x_1)$
…	…	…	…	…
x_j	$\delta(a_1,x_j)$	$\delta(a_2,x_j)$	…	$\delta(a_k,x_j)$
	$\lambda(a_1,x_j)$	$\lambda(a_2,x_j)$	…	$\lambda(a_k,x_j)$

Таблица 1.7.

A / X	a_1	a_2	a_3
x_1	a_2/y_2	a_3/y_3	a_1/y_3
x_2	a_1/y_3	a_1/y_1	a_2/y_2

Таблица 1.8.

Y	y_1	y_2	y_3
A / X	a_1	a_2	a_3
x_1	a_2	a_3	a_1
x_2	a_1	a_1	a_2

Таблица 1.9.

	a_1		a_n
a_1	$x_j(y_k)$		
a_n			$x_j(y_m)$

При графическом способе задания абстрактные автоматы представляются ориентированными графами. Графом автомата называется ориентированный связный граф, вершины которого соответствуют состояниям автомата, а дуги между ними - переходам между состояниями. Две вершины a_k и a_s соединяются дугой в том случае, если в автомате имеется переход из состояния a_k в состояние a_s. Для автомата Мили входной и выходной сигналы проставляются на дуге, соответствующей данному переходу (рис 1.3.), для автомата Мура входной сигнал проставляется на дуге, а выходной - рядом с вершиной, соответствующей состоянию (рис 1.4.).

Здесь рассматриваются только детерминированные автоматы, у которых выполнено условие однозначности переходов: автомат, находящийся в некотором состоянии, под

действием любого входного сигнала не может перейти более чем в одно состояние. Применительно к графическому способу задания автомата это означает, что в графе автомата из любой вершины не могут выходить две и более дуги, отмеченные одним и тем же входным сигналом.

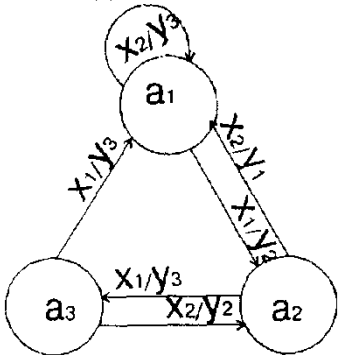


Рисунок 1.3-Граф автомата Мили

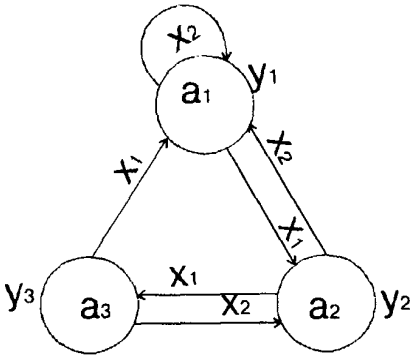


Рисунок 1.4-Граф автомата Мура

При аналитическом способе задания абстрактные автоматы задаются четверкой объектов:

S={ X,A,Y,F}, где F задает для каждого состояния a_j автомата отображение (XxA) -> (AxY). Другими словами, при аналитическом способе задания для каждого состояния автомата a_j указывается отображение F_{ai}, представляющее собой множество всех троек a_p, x_m, y_k, и таких, что под воздействием входного сигнала x_m автомат переходит из состояния a_j в состояние a_p, выдавая при этом выходной сигнал y_k. Применительно к общему определению абстрактного автомата, последнее равнозначно описанию функций δ и λ в соответствии с выражением: $a_p= \delta(a_i, x_m)$, $y_к= \lambda (a_i, x_m)$.

Отображение F_{ai} записывается следующим образом:

$F_{ai}\{a_p(X_m/y_k),a_i(X_f/y_z)\ldots\}$.

Для абстрактного автомата Мили (табл. 1.7.) аналитическое задание имеет следующий вид:

S={ X,A,Y,F}, $X=\{x_1,x_2\}$, $A=\{a_1, a_2, a_3\}$, $Y=\{y_1, y_2, y_3\}$,

$F_{a1}=\{a_2(x_1/y_2), a_1(x_2/y_3)\}$,

$F_{a2}=\{a_3(x_1/y_3), a_1(x_2/y_1)\}$,

$F_{a3}=\{a_1(x_1/y_3), a_2(x_2/y_2)\}$.

Следует отметить, что функция F_{ai} всегда записывается для исходного состояния.

Определим синхронные и асинхронные автоматы. Состояние a_s автомата S называется устойчивым состоянием, если для любого входного сигнала $x_j \in X$, такого, что $a_s= \delta (a_i X_m)$ имеет место $a_s= \delta (a_s, x_m)$.

Автомат S называется асинхронным, если каждое его состояние $a_s \in A$ устойчиво. Автомат S называется синхронным, если он не является асинхронным.

Необходимо заметить, что построенные на практике автоматы всегда асинхронные и устойчивость их состояний всегда обеспечивается тем или иным способом, например, введением сигналов синхронизации. Однако, на уровне абстрактной теории автоматов, когда автомат всего лишь математическая модель, которая не отражает многих конкретных особенностей его реализации, часто оказывается более удобным оперировать с синхронными автоматами.

1.4 Связь между моделями Мили и Мура

Как уже отмечалось, абстрактный автомат работает как преобразователь слов входного алфавита в слова выходного алфавита.

Пусть абстрактный автомат Мили задан графом рис. 1.5.

На вход этого автомата, установленного в начальное состояние, поступает входное слово $X=x_1, x_1, x_2, x_1, x_2, x_2$.

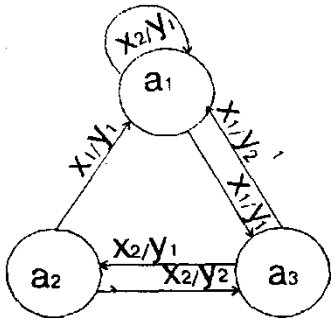


Рисунок 1.5- Граф автомата Мили

Так как $\delta(a_1, x_1) = a_3$, а $\lambda(a_1, x_1) = y_1$, то под действием первой буквы слова X входного сигнала x_1 автомат перейдет в состояние a_3 и на выходе его появится сигнал y_1. Далее, $\delta(a_3, x_1) = a_1$, а $\lambda(a_3, x_1) = y_2$, потому при приходе второго сигнала x_1 автомат окажется в состоянии a_1, а на выходе его появится сигнал y_2. Проследив непосредственно по графу или таблицам переходов и выходов дальнейшее поведение автомата, опишем его тремя строчками, первая из которых соответствует входному слову X, вторая - последовательности состояний, которые проходит автомат под действием букв слова X, третья - выходному слову У, которое появляется на выходе автомата:

x_1	x_1	x_2	x_1	x_2	x_2	
a_1	a_3	a_1	a_1	a_3	a_2	a_3
y_1	y_2	y_1	y_1	y_1	y_2	

Назовем $y = \lambda(a_1, X)$ реакцией автомата Мили в состоянии a_1 на входное слово X. Как видно из примера, в ответ на входное слово длины k автомат Мили выдает последовательность состояний длины к+1 и выходное слово длины k. В общем виде поведение автомата Мили, установленного в состояние a_m, можно описать следующим образом:

Входное слово	x_{i1}	x_{i2}	x_{i3}
Последовательность состояний	a_m	$a_{i2}= \delta(a_m,x_{i1})$	$a_{i3}= \delta(a_{i2},x_{i2})$.
Выходное слово	$y_{i1}= \lambda(a_m,x_{i1})$	$y_{i2}= \lambda(a_{i2},x_{i2})$	$y_{i3}= \lambda(a_{i3},x_{i3})$

Точно так же можно описать поведение автомата Мура, находящегося в состоянии a_m, при приходе входного слова $x_{i1}, x_{i2}, \ldots, x_{ik}$. Напомним, что в соответствии с (1-2) выходной сигнал в автомате Мура в момент времени t (У(t)) зависит лишь от состояния, в котором находится автомат в момент t (a(t)):

Входное слово	x_{i1}	x_{i2}	x_{i3}	
Последовательность состояний	a_m	$a_{i2}= \delta (a_m,x_{i1})$	$a_{i3}= \delta (a_{i2},x_{i2})$	$a_{i4}= \delta (a_{i3},x_{i3})$
Выходное слово	$y_{i1}= \lambda (a_m,x_{i1})$	$y_{i2}= \lambda (a_{i2},x_{i2})$	$y_{i3}= \lambda (a_{i3},x_{i3})$	$y_{i4}= \lambda (a_{i4})$

Очевидно, что выходной сигнал $y_{i1}=\lambda(a_m)$ в момент времени i_1 не зависит от входного сигнала x_{i1}, а определяется только состоянием a_m. Таким образом, этот сигнал y_{i1} никак не связан с входным словом, поступающим на вход автомата, начиная с момента i_1. В связи с этим под реакцией автомата Мура, установленного в состояние a_m на входное слово $X=x_{i1}$, x_{i2}, ..., x_{ik} будем понимать выходное слово той же длины $y= \lambda (a_m, X)=y_{i2}, y_{i3}, \dots, y_{ik+1}$.

В качестве примера рассмотрим автомат Мура S5, граф которого изображен на рис. 1-6, и найдем его реакцию в начальном состоянии на то же самое входное слово которое мы использовали при анализе поведения автомата Мили S1:

Входное слово	x_1	x_1	x_2	x_1	x_2	x_2	
Последовательность состояний	a_1	a_4	a_1	a_1	a_4	a_3	a_5
Выходное слово	y_1	y_1	y_2	y_1	y_1	y_1	y_2

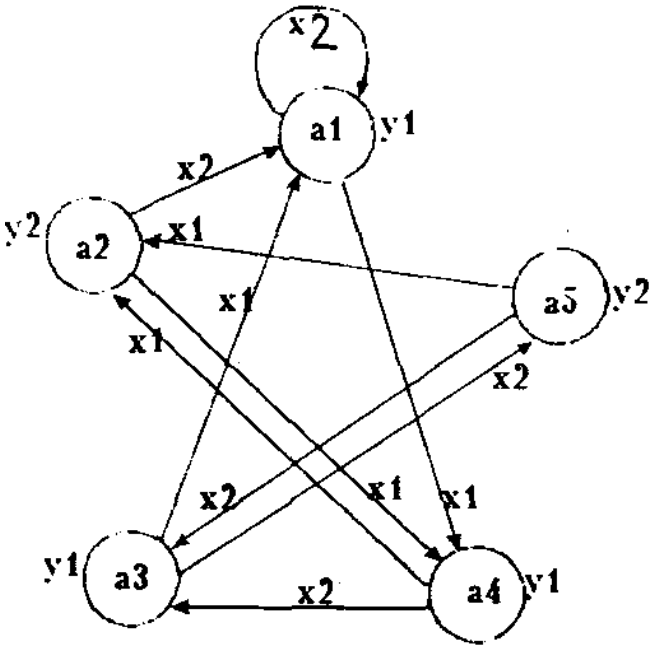


Рисунок 1-6- Граф автомата Мура

Как видно из этого и предыдущего примеров, реакции автоматов S5 и S1 в начальном состоянии на входное слово X с точностью до сдвига на 1 такт совпадают (реакция автомата Мура обведена линией). Дадим теперь строгое определение эквивалентности полностью определенных автоматов.

Два автомата S_A и S_B с одинаковыми входными и выходными алфавитами называются эквивалентными, если после установления их в начальные состояния их реакции на любое входное слово совпадают.

1.5 Эквивалентные автоматы. Эквивалентные преобразования автоматов

Можно показать, что ***для любого автомата Мили существует эквивалентный ему автомат Мура и, обратно, для любого автомата Мура существует эквивалентный ему автомат Мили.***

При описании алгоритмов взаимной трансформации автоматов Мили и Мура в соответствии с изложенным выше мы будем пренебрегать в автоматах Мура выходным сигналом, связанным с начальным состоянием (λ (a_1)).

Рассмотрим сначала преобразование автомата Мура в автомат Мили.

Пусть дан автомат Мура: S_A={ X_A, A_A,$У_A$, δ_A, λ_A,a_{oA}}, где:

X_A={x_1, x_2, ... x_n}; Y={y_1, y_2, ... y_m}; A={ a_0,a_1,a_2,... a_N}; a_{0A} = a_0 – начальное состояние ($a_{0A}\in$ A); δ_A - функция переходов автомата, задающая отображение (X_AxA_A)->A_A; λ_A - функция выходов автомата, задающая отображение A_A->$У_A$.

Построим автомат Мили: S_B={ X_B, A_B, Y_B, δ_B, λ_B, a_{0B}}, у которого A_B=A_A; X_B=X_A; Y_B=$У_A$; δ_B= δ_A; a_{0B}=a_{0A}. Функцию выходов λ_B определим следующим образом. Если в автомате Мура δ_A(a_m, x_1)= a_s и λ_A (a_s)=y_g,то в автомате Мили λ_B (a_m, x_1)=y_g

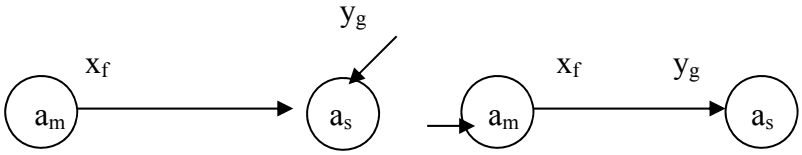


Рисунок 1.7- Иллюстрация перехода от модели Мура к модели Мили

Переход от автомата Мура к автомату Мили при графическом способе задания иллюстрируется рис. 1-7: выходной сигнал y_g записанный рядом с вершиной (a_s), переносится на все дуги, входящие в эту вершину.

При табличном способе задания автомата таблица переходов автомата Мили совпадает с таблицей переходов исходного автомата Мура, а таблица выходов получается из таблицы переходов заменой символа a_s, стоящего на пересечении строки x_f и столбца a_m, символом выходного сигнала y_g отмечающего столбец a_s в таблице переходов автомата S_A.

Из самого способа построения автомата Мили S_B очевидно, что он эквивалентен автомату Мура S_A. По индукции нетрудно показать, что любое входное слово конечной длины, поданное на входы автоматов S_A и S_B, установленных в состояния a_m, вызовет появление одинаковых выходных слов и, следовательно, автоматы S_A и S_B эквивалентны.

Прежде чем рассмотреть *трансформацию автомата Мили в автомат Мура,* наложим на автомат Мили следующее ограничение: у автомата не должно быть преходящих состояний. Под преходящим будем понимать состояние, в которое при представлении автомата в виде графа не входит ни одна дуга, но которое имеет по крайней мере одну выходящую дугу. Итак, пусть задан автомат Мили:

S_A={ X_A, A_A,Y_A, δ_A, λ_A, a_{0A}}, где:

X_A={x_1, x_2, ... x_n}; Y={y_1, y_2, ... y_m}; A={ a_0,a_1,a_2, ... a_N}; a_{0A} = a_0- начальное состояние ($a_{0A}\in$ A); δ_A - функция переходов автомата, задающая отображение (X_AxA_A)->A_A; λ_A - функция выходов автомата, задающая отображение (X_AxA_A)->Y_A.

Построим автомат Мура: S_B={X_B, A_B, Y_B, δ_B, λ_B, a_{0B}}, у которого X_B=X_A; Y_B=Y_A.

Для определения A_B каждому состоянию $a_s\in A_A$ поставим в соответствие множество A_s всевозможных пар вида (a_s, y_g)

Функцию выходов δ_B определим следующим образом. Каждому состоянию автомата Мура S_B, представляющему собой пару вида (a_s,y_g), поставим в соответствие выходной сигнал y_g. Если в автомате Мили S_A был переход δ_A(a_m, x_f)= a_s и при этом выдавался

выходной сигнал λ_A $(a_m,x_f)=y_g$, то в S_B будет переход из множества состояний A_m, порождаемых a_m, в состояние (a_s, y_g) под действием входного сигнала x_f.

В качестве начального состояния a_{0B} можно взять любое из состояний множества A_0, которое порождается начальным состоянием a_0 автомата S_A. При этом выходной сигнал в момент времени t=0 не должен учитываться.

Рассмотрим пример. Пусть задан автомат Мили (табл. 1.10.)

Таблица 10

A	x_1	x_2
a_0	a_2/y_1	a_0/y_1
a_1	a_0/y_1	a_2/y_2
a_3	a_0/y_2	a_1/y_1

Таблица 11

X A	x_1	x_2
a_0 b_0	a_2/y_1 b_{01}	a_0/y_1 b_{02}
a_1	a_0/y_1 b_{11}	a_2/y_2 b_{12}
a_2	a_0/y_2 b_{21}	a_1/y_1 b_{22}

Поставим в соответствие каждой паре a_i/x_k состояние b_{ik} (i-номер состояния, k-номер входного сигнала), с учетом b_0.

Составим таблицу переходов автомата Мура, руководствуясь следующими правилами:

1) Выпишем из таблицы 1.11 состояния автомата Мили и соответствующие каждому из них множества состояний автомата Мура (b_{ik}):

$a_0= \{b_0, b_{02}, b_{11}, b_{21}\}$; $a_1= \{b_{22}\}$; $a_2= \{b_{01}, b_{12}\}$;

2) Если состояние автомата Мура b_{ik} входит в множество, соответствующее состоянию a_p автомата Мили, то в строку таблицы переходов автомата Мура для состояния b_{ik} следует записать строку из таблицы переходов автомата Мили, соответствующую состоянию a_p (из 1.10.).

3) Функцию выходов автомата Мура определим следующим образом: $\lambda_B(b_{ik})= \lambda_A(a_i, x_K)$. Для начального состояния b_0 значение выходного сигнала можно выбрать произвольно, но порождаемый начальным состоянием a_0 (с учетом понятия эквивалентности состояний). Результирующая таблица переходов и выходов автомата Мура эквивалентного автомату Мили, заданному таблицей 1.10. представлена в таблице 1.12.

4) Найдем в таблице 1.12. эквивалентные состояния и удалим их (заменим на представителя класса эквивалентности).

Если выходной сигнал возле b_0 доопределить y_1, то окажется, что в данной таблице переходов находится 3 эквивалентных состояния (b_0,b_{11},b_{02}). Заменив класс эквивалентности одним представителем (b0), получим окончательную таблицу переходов (табл. 1.13)

Таблица 1.12.

	x_1	x_2	Y
b_0	b_{01}	b_{02}	y_1
b_{01}	b_{21}	b_{22}	y_1
b_{02}	b_{01}	b_{02}	y_1
b_{11}	b_{01}	b_{02}	y_1
b_{12}	b_{21}	b_{22}	y_2
b_{21}	b_{01}	b_{02}	y_2
b_{22}	b_{11}	b_{12}	y_1

Таблица 1.13.

	x_1	x_2	y
b_0	b_{01}	b_0	y_1
b_{01}	b_{21}	b_{22}	y_1
b_{12}	b_{21}	b_{22}	y_2
b_{21}	b_{01}	b_0	y_2
b_{22}	b_0	b_{12}	y_1

Изложенные методы взаимной трансформации автоматов Мили и Мура показывают, что при переходе от автомата Мура к автомату Мили число состояний автомата не изменяется, тогда как при обратном переходе число состояний в автомате Мура, как правило, возрастает.

Таким образом, эквивалентные между собой автоматы могут иметь различное число состояний, в связи с чем возникает задача нахождения минимального (с минимальным числом состояний) автомата в классе эквивалентных между собой автоматов. Существование для любого абстрактного автомата эквивалентного ему абстрактного автомата с минимальным числом внутренних состояний впервые было доказано Муром.

1.6 Минимизация числа внутренних состояний автомата. Алгоритм Ауфенкампа-Хона

В основу метода минимизации состояний автомата положена идея разбиения всех состояний исходного, абстрактного автомата на попарно не пересекающиеся классы эквивалентных состояний и замене каждого класса эквивалентности одним состоянием (представителем данного класса). Образующийся в результате этих преобразований минимальный автомат имеет столько же состояний, на сколько классов эквивалентности разбиваются исходные состояния.

Два состояния автомата a_m и a_s называются эквивалентными ($a_m \equiv a_s$), если $\lambda(a_m,X) = \lambda(a_s,X)$ для всех возможных входных слов длины X.

Если a_m и a_s не эквивалентны, они различимы. Более слабой эквивалентностью является K-эквивалентность. Состояния a_m и a_s K-эквивалентны, если $\lambda(a_m, X_k) = \lambda(a_s,X_k)$ для всех возможных входных слов длины К. При минимизации числа внутренних состояний автомата Мили $S=\{X, Y, A, \delta, \lambda, a_0\}$ используется алгоритм Ауфенкампа-Хона:

1. Находят последовательные разбиения π_1, π_2,…, π_k, π_{k+1}, множества A на классы одно-, двух-, ..., К-, (К+1) -эквивалентных состояний до тех пор, пока на каком-то (К+1) шаге не окажется, что $\pi_k = \pi_{k+1}$. В этом случае К-эквивалентные состояния являются эквивалентными. Число шагов К, при котором $\pi_k = \pi_{k+1}$ не превышает N-1, где N- число внутренних состояний автомата.

2. В каждом классе эквивалентности π выбирают по одному элементу (представителю класса), которые образуют множества A' состояний минимального автомата S'.

3. Функцию переходов δ' и выходов λ' автомата S' определяют на множестве A'xX. Для этого в таблице переходов и выходов вычеркивают столбцы, соответствующие не вошедшим в множество A' состояниям, а в оставшихся столбцах таблицы переходов все состояния заменяются на эквивалентные из множества A', (на представителей).

4. В качестве a'_0 выбирается одно из состояний, эквивалентное состоянию a_0. В частности, удобно принять само состояние a_0.

При минимизации автомата Мура вводится понятие 0-эквивалентности состояний и разбиения множества состояний на 0-классы: 0-эквивалентными называются любые, одинаково отмеченные выходными сигналами, состояния автомата Мура.

В качестве примера рассмотрим минимизацию автомата Мура, заданного таблицей переходов и выходов (Таблица 1.14).

Таблица 1.14

У	y_1	y_1	y_3	y_3	y_3	y_2	y_3	y_1	y_2	y_2	y_2	y_2
A	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}
x_2	a_{10}	a_{12}	a_5	a_7	a_3	a_7	a_3	a_{10}	a_7	a_1	a_5	a_2
x_2	a5	a7	a6	a_{11}	a_9	a_{11}	a_6	a_4	a_6	a_8	a_9	a_8

Выполним разбиение π_0:

π_0={B_1,B_2,B_3};

B_1={a_1, a_2, a_8}, B_2={a_6, a_9, a_{10}, a_{11}, a_{12}}, B_3={a_3, a_4, a_5, a_7}. Построим таблицу разбиения π_0:

У	B_1		B_2						B_3			
A	a_1	a_2	a_8	a_6	a_9	a_{10}	a_{11}	a_{12}	a_3	a_4	a_5	a_7
x_1	B_2	B_2	B_2	B_3	B_3	B_1	B_3	B_1	B_3	B_3	B_3,	B_3
x_2	B_3	B_3	B_3	B_2	B_2	B_1	B_2	B_1	B_2	B_2	B_2	B_2

Выполним разбиение π_1:

π_1={C_1,C_2,C_3,C_4};

C_1={a_1, a_2, a_8}, C_2={a_6, a_9, a_{11}}, C_3={ a_{10}, a_{12}}, C_4={a_3, a_4, a_5, a_7}.

Построим таблицу разбиения π_1:

У	C_1			C_2			C_3		C_4			
A	a_1	a_3	a_8	a_6	a_9	a_{11}	a_{10}	a_{12}	a_3	a_4	a_5	a_7
x_1	C_3	C_3	C_3	C_4	C_4	C_4	C_1	C_1	C_4	C_4	C_4	C_4
x_2	C_4	C_4	C_4	C_2	C_2	C_2	C_1	C_1	C_2	C_2	C_2	C_2

Выполним разбиение π_2.

π_1={D_1, D_2, D_3, D_4};

D_1={a_1, a_2, a_8}, D_2={a_6, a_9, a_{11}}, D_3={ a_{10}, a_{12}}, D_4={a_3, a_4, a_5, a_7}.

Разбиение π_2 повторяет разбиение π_1 - процедура разбиения завершена.

Выберем произвольно из каждого класса эквивалентности D_1, D_2, D_3, D_4 по одному представителю - в данном случае по минимальному номеру: A'={a_1, a_3, a_6, a_{10}}.

Удаляя из исходной таблицы переходов "лишние" состояния, определяем минимальный автомат Мура (табл. 1.15.)

Таблица 1.15.

У	y_1	y_3	y_2	y_2
A	a_1	a_3	a_6	a_{10}
x_1	a_{10}	a_3	a_3	a_1
x_2	a_3	a_6	a_6	a_1

2 СТРУКТУРНЫЕ АВТОМАТЫ

2.1 Основные понятия. Канонический метод структурного синтеза автоматов. Теорема Глушкова о структурной полноте

Вслед за этапом абстрактного синтеза автоматов, заканчивающимся минимизацией числа состояний, следует этап структурного синтеза, *цепью которого является построение схемы, реализующей автомат из логических элементов заданного типа.*

Если абстрактный автомат был лишь математической моделью дискретной системы, то в структурном автомате учитывается структура входных и выходных сигналов автомата, а также его внутреннее устройство на уровне структурных схем. Структурным синтезом занимается структурная теория автоматов, основной задачей которой является нахождение общих приемов построения структурных схем автоматов на основе композиции элементарных автоматов, принадлежащих к заранее заданному конечному числу типов.

Под композицией элементарных автоматов в общем случае понимается следующее.

Пусть заданы элементарные автоматы $S_1,...,S_k$. Произведем объединение элементарных автоматов в систему совместно работающих автоматов. Введем в рассмотрение некоторое конечное множество узлов, называемых внешними входными и внешними выходными узлами. Эти узлы отличаются от узлов рассматриваемых элементарных автоматов, которые носят название внутренних. Композиция автомата состоит в том, что в полученной системе элементарных автоматов $S_1,...,S_k$ и внешних узлов производится отождествление некоторых узлов (как внешних, так и внутренних). С точки зрения совместной работы системы автоматов смысл операции отождествления узлов состоит в том, что элементарный сигнал, попадающий на один из узлов, входящих в множество отождествляемых между собой узлов, попадает тем самым на все узлы этого множества, После проведенных отождествлений узлов система автоматов превращается в так называемую схему (сеть) автоматов. Будем считать, что автоматы, входящие в схему автоматов, работают совместно, если в каждый момент автоматного времени на все внешние входные узлы подается набор входных сигналов (структурный входной сигнал схемы) и со всех внешних выходных узлов снимается набор выходных сигналов (структурный выходной сигнал).

В структурной теории как входные так и выходные каналы считаются состоящими из элементарных входных (выходных) каналов. По всем элементарным входным (выходным) каналам могут передаваться только элементарные сигналы.

Рисунок 2.1- Структурный автомат

Набор возможных значений сигналов, подаваемых на один внешний входной (выходной) узел, называется структурным входным (выходным) алфавитом автомата. Алфавит должен быть конечным.

Входной и выходной сигналы задаются конечными упорядоченными наборами элементарных сигналов, называемыми векторами, а составляющие их элементарные сигналы - компоненты векторов. Число компонентов вектора - это размерность алфавита.

Например, $X=\{x_1, x_2, x_3, x_4, x_5\}$ - входной алфавит абстрактного автомата.

Структурный входной алфавит, размерность которого равна трем:

X_1=000, x_2=001, x_3=010, x_4=011, x_5= 100.

Векторное представление входных и выходных сигналов называется структурным входным выходным сигналом, соответственно.

Предполагается, что все входящие в композицию автоматы имеют один и тот же структурный алфавит и работают в одном и том же автоматном времени. В настоящее время наиболее распространенным структурным алфавитом является двоичный, что объясняется простотой его представления в современных элементах и приборах. Кроме того, для двоичного алфавита наиболее разработан аппарат булевых функций, позволяющий производить многие операции над схемой формально. Поэтому в дальнейшем при решении задач структурного синтеза автоматов будет использоваться в основном двоичный, структурный алфавит.

Предположим, что в каждый момент автоматного времени структурный выходной сигнал схемы однозначно определяется поступившей к этому времени конечной последовательностью структурных входных сигналов, начальными состояниями входящих в схему автоматов и сделанными при построении схемы отождествлениями узлов. В этом случае построенную схему будем рассматривать как некоторый автомат S, а саму схему назовем структурной схемой этого автомата.

Построенный таким образом автомат S есть результат композиции элементарных автоматов S_1,...,S_k. В отличие от абстрактного С-автомата, имеющего один входной и два выходных канала, на которые поступают сигналы во входном и выходных алфавитах автомата, структурный автомат имеет входные и выходные каналы, на которых появляются сигналы в структурном алфавите автомата. В случае двоичного алфавита каждый входной и выходные сигналы абстрактного автомата могут быть закодированы векторами различной длины соответственно, компоненты которых принимают два значения - нуль и единицу.

На этапе структурного синтеза предварительно выбираются элементарные автоматы, из которых затем путем их композиции строится структурная схема полученного на этапе абстрактного синтеза автомата Мили, Мура или С-автомата. Если решение задачи структурного синтеза существует, говорят, что заданная система автоматов структурно полна.

В настоящее время нет сколько-нибудь эффективных методов (существенно более простых, чем метод перебора всех вариантов) решения основной задачи структурного синтеза при любом наборе структурно полных систем элементарных автоматов. Поэтому обычно применяется так называемый канонический метод структурного синтеза, при котором используются элементарные автоматы некоторого специального вида: автоматы с памятью, имеющие более одного состояния, и автоматы без памяти - с одним состоянием. Автоматы первого класса носят название элементов памяти, а автоматы второго класса - комбинационных или логических элементов.

Теоретическим обоснованием канонического метода структурного синтеза автоматов является доказанная в теорема о структурной полноте (теорема Глушкова):

Всякая система элементарных автоматов, которая содержит автомат Мура с нетривиальной памятью, обладающий полной системой переходов и полной системой выходов, и какую-либо функционально полную систему логических элементов, является структурно полной.

Существует общий конструктивный прием (канонический метод структурного синтеза), позволяющий в рассматриваемом случае свести задачу структурного синтеза произвольных автоматов к задаче синтеза комбинационных схем.

Результатом канонического метода структурного синтеза является система логических уравнений, выражающая зависимость выходных сигналов автомата (функции выходов автомата) и сигналов, подаваемых на входы запоминающих элементов, от сигналов, приходящих на вход всего автомата в целом, и сигналов, снимаемых с выхода элементов

памяти (функции возбуждения элементов памяти автомата). Эти уравнения называются ***каноническими.***

Для правильной работы схем, очевидно, нельзя разрешать, чтобы сигналы на входе запоминающих элементов непосредственно участвовали в образовании выходных сигналов, которые по цепям обратной связи подавались бы в тот же самый момент времени на эти входы. В связи с этим запоминающими элементами должны быть не автоматы Мили, а автоматы Мура (см. уравнения функционирования этих автоматов).

Таким образом, структурно полная система элементарных автоматов должна содержать хотя бы один автомат Мура. В то же время для синтеза любых автоматов с минимальным числом элементов памяти необходимо в качестве таких элементов выбирать автоматы Мура, имеющие полную систему переходов и полную систему выходов - так называемые полные автоматы.

Рассмотрим полноту автоматов памяти на примере автомата Мура. (табл. 2.1.) Полнота системы переходов означает, что для любой пары состояний (a_m,…,a_s) автомата найдется входной сигнал, переводящий первый элемент этой пары a_m во второй - a_s т. е. в таком автомате в каждом столбце таблицы переходов должны встречаться все состояния автомата. Полнота системы выходов автомата Мура состоит в том, что каждому состоянию автомата поставлен в соответствие свой особый выходной сигнал, отличный от выходных сигналов других состояний.

Таблица 2.1.

У	y_1	y_2	y_3
A\X	x_1	x_2	x_3
a_1	a_1	a_2	a_3
a_2	a_3	a_1	a_2
a_3	a_2	a_3	a_1

Очевидно, что в таком автомате число выходных сигналов равно числу состояний автомата. Вместе с предыдущим утверждением это приводит к тому, что в автомате Мура с полной системой выходов можно отождествить состояния автомата с его выходными сигналами. В связи с этим в автоматах памяти мы будем использовать одни и те же обозначения и для состояний, и для выходных сигналов, т. е. отмеченная таблица переходов в автоматах Мура с полной системой выходов превращается просто в таблицу переходов. Автомат Мура в табл. 2.1. удовлетворяет условиям полноты системы переходов и выходов.

Наличие функционально полной системы логических элементов позволяет реализовать булеву функцию любой степени сложности.

После выбора элементов памяти и кодирования состояний синтез структурного автомата сводится к синтезу комбинационной схемы, реализующей канонические уравнения.

Автомат памяти также можно рассматривать на абстрактном и структурном уровнях. Абстрактный автомат памяти имеет один входной и один выходной каналы. При переходе от абстрактного автомата к структурному автомату входные и выходные сигналы должны быть закодированы соответствующими двоичными наборами.

2.2 Основные этапы канонического метода структурного синтеза

В каноническом методе структурного синтеза можно выделить несколько основных этапов:

1. Кодирование алфавитов автомата.
2. Выбор элементов памяти.
3. Выбор функционально полной системы логических элементов.
4. Запись и минимизация канонических уравнений.
5. Построение функциональной схемы автомата.

Исходными данными для начала работы данного метода являются абстрактный цифровой автомат с памятью, заданный таблицей переходов и выходов. Рассмотрим подробнее каждый из этапов канонического метода.

2.2.1 Кодирование алфавитов автомата

На структурном уровне каждая буква входного алфавита $x \in X$, каждая буква выходного алфавита $y \in Y$ и каждая буква алфавита состояний $a \in A$ кодируется двоичными векторами (двоичными наборами), число компонент которых определяется следующим образом:

$K_{вх} >= int(\log_2|X|)$; $K_{вых} >= int(\log_2|Y|)$; $K_{сост} >= int(\log_2|A|)$;

где int - ближайшее большее целое число.

|X|, |У|, |A| - мощность алфавита входного, выходного и состояний, соответственно. Мощностью алфавита называется количество различных символов входящих в этот алфавит.

Процесс замены буква алфавита (X, У, A) абстрактного автомата двоичными векторами носит название кодирования и описывается таблицами кодирования. Таблица кодирования имеем следующий вид: в левой части перечисляются все символы абстрактного алфавита, а в правой - соответствующие им двоичные векторы.

Рассмотрим пример.

Абстрактный автомат Мили задан таблицей переходов и выходов (табл. 2.2.). Выполнить кодирование алфавитов автомата.

Таблица 2.2

A\ X	x_1	x_2
a_1	a_2/y_1	a_3/y_3
a_2	a_1/y_2	a_2/y_1
a_3	a_2/y_2	a_1/y_1

Выпишем алфавиты автомата и определим длины векторов кодов алфавитов:

$X=\{x_1,x_2\}$; $K_{вх} >= int(\log_2|2|)=1$,

$Y=\{y_1,y_2,y_3\}$; $K_{вых} >= int(\log_2|3|)=2$,

$A=\{a1,a2,a3\}$; $K_{сост} >= int(\log_2|3|)=2$.

Заполним таблицы кодирования:

Таблица 2.3

x_1	0
x_2	1

Таблица 2.4

У	
y_1	00
y_2	01
y_3	10

Таблица 2.5

A	
a_1	00
a_2	01
a_3	10

Результирующая таблица - структурная таблица переходов и выходов автомата (табл. 2.6.) Получением структурной таблицы переходов-выходов автомата заканчивается этап кодирования.

Таблица 2.6.

A\ X	0	1
00	01/00	10/10
01	00/01	01/00
10	01/01	00/00

В общем случае, каждой кодируемой букве может быть присвоен произвольный двоичный вектор, но обязательно две различные буквы одного алфавита должны кодироваться различными векторами. Коды, соответствующие символам различных алфавитов могут совпадать, в рассмотренном примере - коды выходных сигналов и состояний.

На данном этапе целесообразно отметить, что способ кодирования состояний в значительной степени определяют сложность реализации комбинационных схем. Существуют специальные способы кодирования, обеспечивающие получение экономичных схем. Они будут рассмотрены ниже.

2.2.2 Выбор элементов памяти автомата

Замена таблицы переходов автомата на структурную таблицу переходов приводит к тому, что функция переходов $\delta(a_i,x_j)$ автомата становится векторной. Иными словами, аргументами такой функции являются все возможные пары двоичных векторов (a_i,x_j), а сама функция принимает значение из множества A двоичных векторов состояний автомата. В соответствии со структурной таблицей переходов автомата его векторная функция переходов каждой паре двоичных векторов ставит в соответствие определенный двоичный вектор a_k, что на абстрактном уровне определяется соотношением $a_k = \delta(a_i,x_j)$. Из этого следует, что структурный автомат должен запоминать двоичный вектор каждого очередного состояния автомата, для чего служат элементы памяти (запоминающие элементы, триггеры).

При каноническом методе структурного синтеза автоматов в качестве элементов памяти используются элементарные автоматы Мура с двумя состояниями, обладающие полной системой переходов и выходов.

Полнота системы переходов автомата в общем случае означает, что для любой пары состояний автомата существует входной сигнал, переводящий элементарный автомат из одного состояния в другое. Таблица переходов элементарного автомата с полной системой переходов должна содержать в каждой своей строке все возможные состояния.

Полнота системы выходов означает, что различным состояниям автомата соответствуют различные выходные сигналы. Обычно нулевому состоянию элементарного автомата соответствует нулевой выходной сигнал, единичному - единичный.

Очевидно, что число элементов памяти структурного автомата равно числу компонент вектора его состояний.

2.2.3 Выбор функционально-полной системы логических элементов

Функционирование структурного автомата во времени предполагает управление переключением каждого элементарного автомата его памяти в соответствии со структурной таблицей переходов синтезируемого автомата. Последнее осуществляется с помощью специальной комбинационной схемы, подключаемой к информационным входам элементарного автомата памяти и реализующей булевы функции, управляющие его переключением.

Такие булевы функции называются функциями возбуждения элемента памяти и, в общем случае, различных функций возбуждения столько, сколько различных информационных входов имеется у элементарных автоматов памяти в синтезируемом структурном автомате.

Функция возбуждения любого элемента памяти является произвольной булевой функцией и для ее реализации комбинационной схемой необходимо использовать какую-либо функционально-полную систему логических элементов. Теоретическим фундаментом канонического метода структурного синтеза автоматов с памятью является теорема о структурной полноте, из которой следует, что для построения структурного автомата необходимо кроме элементов памяти иметь комбинационную схему, реализующую булевы функции возбуждения элементов памяти автомата, а для выработки выходных сигналов структурного автомата - специальную комбинационную схему формирования выходных сигналов автомата.

Функция возбуждения структурного автомата является векторной. Ее аргументами являются пары двоичных векторов (a_i,x_j).- а значением функции - двоичный вектор, каждая i-я компонента которого есть значение булевой функции возбуждения i-го элемента памяти автомата, определяющая тот двоичный сигнал, который необходимо подать на вход i-го элемента памяти для обеспечения его переключения в соответствии с требованиями структурной таблицы переходов.

Если векторная функция переходов задает переход из одного вектора состояния структурного автомата в другой вектор состояния под воздействием двоичного вектора входного сигнала, то векторная функция возбуждения автомата задает двоичный вектор, который нужно подать на входы элементов памяти автомата, чтобы обеспечить требуемый переход (в соответствии с векторной функцией переходов автомата).

Последнее означает, что переменными, от которых зависит векторная функция возбуждения, являются те же переменные, что и для векторной функции переходов автомата, т.е. выходы всех элементов памяти автомата и входы автомата. Поэтому структурный автомат Мура, в общем случае, может быть представлен структурной схемой (рис. 2.2), а структурный автомат Мили - структурной схемой (рис. 2.3).

Буквами $\alpha_1,... \alpha_\varphi$ на рисунках обозначены выходы элементов памяти. Буквами $\varphi_1,\ldots, \varphi_j$ обозначены булевы функции возбуждения элементов памяти. Для простоты положим, что каждый элемент памяти структурного автомата имеет один информационный вход. Буквами $w_1,...,w_j$ обозначены выходные каналы автомата, где j - число выходных каналов автомата. Буквами $z_1,\ldots,z_m$ – входные каналы.

2.2.4 Построение уравнений булевых функций возбуждения и выходов автомата

Кодирование и выбор системы элементов однозначно определяют комбинационную часть автомата: вначале строится таблица истинности функций возбуждения элементов памяти автомата, получившая название таблицы функций возбуждения; канонические уравнения функций возбуждения выписываются исходя из построенной таблицы. Полученная аналитическая запись булевой функции возбуждения (для каждого элемента памяти автомата) может быть минимизирована любым из известных методов. Исходными данными для построения таблицы функций возбуждения являются структурная таблица переходов автомата и таблица переходов элемента памяти. Обрамление таблицы функций возбуждения, т.е. идентификация ее строк и столбцов полностью совпадает с обрамлением структурной таблицы переходов синтезируемого автомата. Клетки, расположенные внутри таблицы функций возбуждения, заполняются специальным образом.

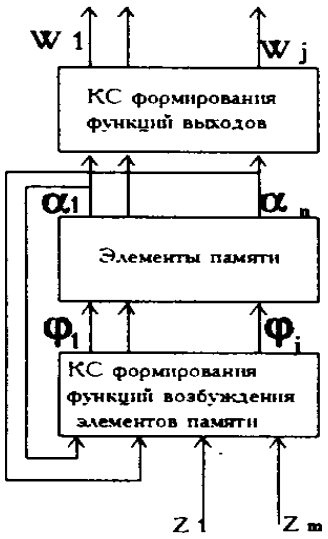


Рисунок 2.2 - Структурная схема автомата Мура

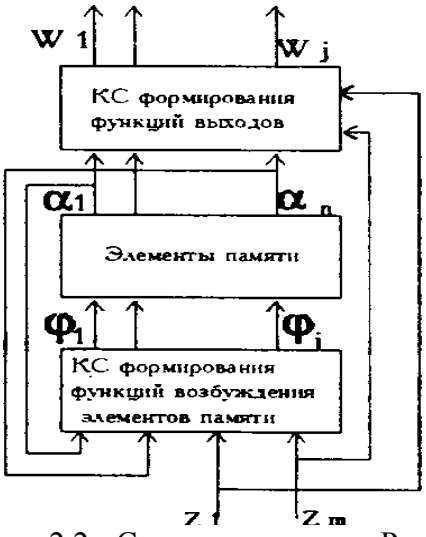


Рисунок 2.3 - Структурная схема автомата Мили

Получение канонических уравнений булевых функций выходов структурного автомата проще и может быть сделано непосредственно по структурной таблице выходов автомата. Структурная таблица выходов автомата есть таблица истинности булевых функций выходов автомата. Иными словами, уравнения булевых функций выходов автомата не зависят от типа используемых элементов памяти, однако зависят от их количества.

Если синтезируемый автомат является автоматом Мура, то задача построения уравнений булевых функций возбуждения решается так же. Уравнения булевых функций выходов автомата Мили строятся несколько иначе. Последнее связано с различиями в способах построения структурных таблиц выходов автомата Мили и Мура. После проведения этапа кодирования состояний автомата и кодирования выходных сигналов получаем структурную таблицу выходов автомата, которая является таблицей истинности булевых функций выходов автомата.

2.2.5 Построение функциональной схемы автомата

На основании полученных выражений для булевых функций возбуждения элементов памяти автомата и булевых функций выходов автомата строятся комбинационная схема функций возбуждения и комбинационная схема формирования выходных сигналов автомата. Элементы памяти подключаются к построенным комбинационным схемам. Функциональная схема автомата Мура отлична только комбинационной схемой формирования выходных сигналов, которая строится на основании уравнений для булевых функций выходов. Отметим, что реализация комбинационных схем может быть выполнена в любом функционально-полном базисе.

2.3 Пример канонического метода структурного синтеза

Пусть задан абстрактный автомат Мили таблицей переходов и выходов (табл. 2.7.). Используя логические элементы И, ИЛИ, НЕ и элементарный автомат Мура (элемент памяти), заданный табл. 2.8.

Таблица 2.7.

Таблица 2.8

A\ X	x_1	x_2	x_3
a_1	a_2/y_3	a_3/y_4	a_2/y_2
a_2	-	a_1/y_3	a_3/y_1
a_3	a_1/y_2	-	a_3/y_3

	r_1	r_2
b_1	b_1	b_2
b_2	b_2	b_1

Выполним кодирование алфавитов автомата (табл. 2.7.)

Выпишем алфавиты автомата и определим длины векторов кодов алфавитов:

X = {x_1, x_2, x_3}; $K_{вх} >= int(\log_2|3|)= 2$,
У = {y_1, y_2, y_3, y_4}; $K_{вых} >= int(\log_2|3|)= 2$,
A = {a_1, a_2, a_3}; $K_{сост} >= int(\log_2|3|)= 2$.

Заполним таблицы кодирования (табл. 2.9 – 2.11):

Таблица 2.9.

X	z_1	z_2
x_1	0	0
x_2	0	1
x_3	1	0

Таблица 2.10.

У	w_1	w_2
y_1	1	0
y_2	0	0
y_3	1	1
y_4	0	1

Таблица 2.11.

A	α_1	α_2
a_1	0	0
a_2	0	1
a_3	1	1

Каждый разряд вектора кода обозначим символом с соответствующим номером.

Входные - z, выходные - w, состояния – α.

Таблица 2.12

	z_1z_2		
$\alpha_1\ \alpha_2$	00	01	10
00	01/11	11/01	01/00
01	-	00/11	11/10
11	00/00	-	11/11

В результате получим таблицу переходов и выходов структурного автомата (табл. 2.12.). Выполним кодирование элементарного автомата Мура (табл. 2.8.):

- выпишем алфавиты автомата и определим длины векторов кодов алфавитов,

R={r_1,r_2}; $K_{вх} >= int(\log_2|2|)=1$,
B = {b_1, b_2}; $K_{сост} >= int(\log_2|2|)= 1$;

-заполним таблицы кодирования:

Таблица 2.13.

R	φ
r_1	0
r_2	1

Таблица 2.14.

B	β
b_1	0
b_2	1

Структурная таблица переходов элементарного автомата Мура имеет вид (табл. 2.15.):

Таблица 2.15.

	0	1
0	0	1
1	1	0

Так как абстрактный автомат имеет три состояния, каждое из которых кодируется двумя разрядами, то структурный автомат будет содержать два запоминающих элемента.

Теперь задача сводится к синтезу комбинационной схемы, реализующей канонические уравнения:

$w_1=\lambda(\alpha_1, \alpha_2, z_1, z_2)$, $w_2= \lambda(\alpha_1, \alpha_2, z_1, z_2)$ - функции выходов автомата

$\varphi_1= f(\alpha_1, \alpha_2, z_1, z_2)$, $\varphi_2= f\, \alpha_1, \alpha_2, z_1, z_2)$ - функции возбуждения элементов памяти автомата.

Функции w_1, w_2 можно получить непосредственно из отмеченной таблицы переходов структурного автомата как дизъюнкцию конъюнкций, соответствующих тем наборам $(\alpha_1,\alpha_2,z_1,z_2)$, на которых эти функции принимают значения 1. Но более удобно пользоваться так называемой таблицей формирования функций возбуждения и функций выходов автомата, в которой в табличной форме задана система булевых функций (табл. 2.16). Заполним эту таблицу, используя коды соответствующих алфавитов и таблицу переходов и выходов абстрактного автомата. Для заполнения колонок φ_1, φ_2 необходимо воспользоваться еще и таблицей элемента памяти (табл. 2.15).

Таблица 2.16.

Исходное состояние		Входной сигнал		Состояние перехода		Функции возбуждения эл-в памяти		Выходной сигнал	
α_1	α_2	z_1	z_2	α_1	α_2	φ_1	φ_2	w_1	w_2
0	0	0	0	0	1	0	1	1	1
		0	1	1	1	1	1	0	1
		1	0	0	1	0	1	0	0
0	1	0	1	0	0	0	1	1	1
		1	0	1	1	1	0	1	0
1	1	0	0	0	0	1	1	0	0
		1	0	1	1	0	0	1	1

Для заполнения функций возбуждения элементов памяти рассматривается переход из исходного состояния (α_1 α_2) в состояние перехода (α_1 α_2). За первый разряд α_1 отвечает первый элемент памяти (его функция φ_1), за второй α_2 - второй элемент памяти (его функция φ_2).

В таблице проставляется значение входного сигнала, который обеспечивает соответствующий переход. Количество функций возбуждения элементов памяти автомата зависит от количества разрядов вектора кода состояния и от количества информационных входов самого запоминающего элемента. Рассмотрим, например, что будет со структурным автоматом, если он находится в состоянии 01, и на его вход поступил сигнал 10.

Как видно из таблицы переходов структурный автомат перейдет в состояние 11. Этот переход складывается из двух переходов элементов памяти: 1-й из 0 в 1, 2-й из 1 в 1. По таблице 2.15. определим входные сигналы элемента памяти, обеспечивающие эти переходы: это 0 и 1., и т.д. В клетку соответствующего перехода запишем вектор функции возбуждения, вызывающий данный переход.

По таблице 2.16 запишем аналитические выражения канонических уравнений:

$$\varphi_1 = \bar{a}_1\bar{a}_2\bar{z}_1 z_2 \vee \bar{a}_1 a_2 z_1 \bar{z}_2 \vee a_1 a_2 \bar{z}_1 \bar{z}_2 = 1 \vee 6 \vee 12$$

$$\varphi_2 = \bar{a}_1\bar{a}_2\bar{z}_1\bar{z}_2 \vee \bar{a}_1\bar{a}_2 z_1 z_2 \vee \bar{a}_1\bar{a}_2 z_1\bar{z}_2 \vee \bar{a}_1 a_2\bar{z}_1 z_2 \vee a_1 a_2\bar{z}_1\bar{z}_2 = 0 \vee 1 \vee 2 \vee 5 \vee 12$$

$$w_1 = \bar{a}_1\bar{a}_2\bar{z}_1\bar{z}_2 \vee \bar{a}_1 a_2\bar{z}_1 z_2 \vee \bar{a}_1 a_2 z_1\bar{z}_2 \vee a_1 a_2 z_1\bar{z}_2 = 0 \vee 5 \vee 6 \vee 14$$

$$w_2 = \bar{a}_1\bar{a}_2\bar{z}_1\bar{z}_2 \vee \bar{a}_1\bar{a}_2 z_1 z_2 \vee \bar{a}_1 a_2\bar{z}_1 z_2 \vee a_1 a_2 z_1\bar{z}_2 = 0 \vee 1 \vee 5 \vee 14$$

Структурная схема данного автомата приведена на рис.2.4.

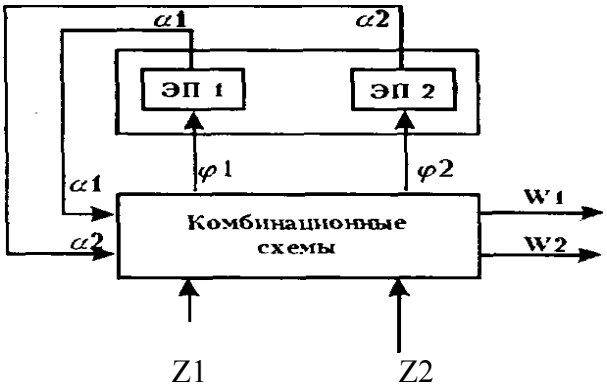


Рисунок 2.4-Структурная схема автомата

Не занимаясь минимизацией канонический уравнений, построим схему электрическую функциональную (рис. 2.5).

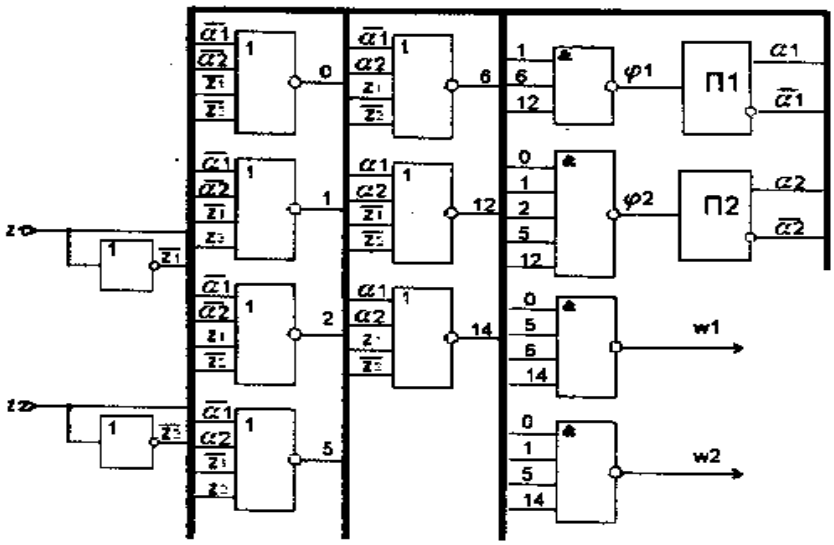


Рисунок 2.5- Функциональная схема автомата

Рассмотрим некоторые из этапов канонического метода более подробно, с применением специальных методов.

2.4 Элементы памяти

В качестве элементов памяти структурного автомата обычно используются триггеры. Как уже было сказано, *с* точки зрения прикладной теории цифровых автоматов, триггер - это элементарный автомат Мура, обладающий полной системой переходов и полной системой выходов.

Триггер характеризуется числом информационных входов, внутренних состояний, числом выходных сигналов и т.д. выходные сигналы триггера отождествляются с его внутренними состояниями, именно поэтому таблица переходов совпадает с таблицей выходов и триггер задается только одной из них (таблицей переходов). Как правило, триггер формирует как прямой сигнал, так и инверсный.

2.4.1 Элементы памяти с одним информационным входом

Существует только 4 типа запоминающих элементов с одним информационным входом, имеющих полную систему переходов и выходов: D-триггер, T-триггер, $\overline{D}$, -триггер, $\overline{T}$ -триггер. Таблицы их переходов представлены табл. 2.17 - 2.20. соответственно, а условные графические изображения триггеров представлены на рис. 2.6. Входы D, T, $\overline{D}$, $\overline{T}$ называются информационными.

Таблицы переходов триггеров составляются только для информационных входов. Остальные входы являются вспомогательными. В частности, вход C - вход для подключения синхросерии (о чем будет сказано ниже). Каждый из триггеров имеет два выхода. Появление единичного сигнала на выходе, помеченном на рисунках символом q, означает, что триггер находится в единичном состоянии. Появление единичного сигнала на выходе $\overline{q}$ говорит о нулевом состоянии.

Таблица 2.17.

D	Q(t)	Q(t+1
0	0	0
0	1	0
1	0	1
1	1	1

Таблица 2.18

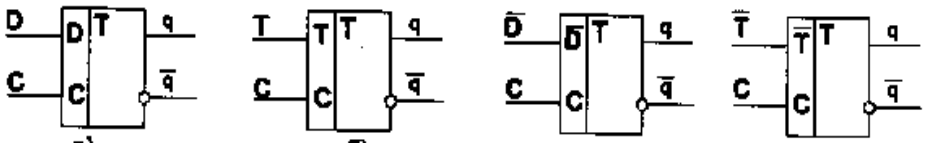
	T	Q(t)	Q(t+1)
	0	0	0
	0	1	1
	1	0	1
	1	1	0

Таблица 2.19

D	Q(t)	Q(t+1)
0	0	1
0	1	1
1	0	0
1	1	0

Таблица 2.20.

T	Q(t)	Q(t+1)
0	0	1
0	1	0
1	0	0
1	1	1



а б в г

Рисунок 2.6- Условное графическое обозначение триггеров:
а) D-триггер; б) Т-триггер; в) $\overline{D}$-триггер; г) $\overline{T}$ -триггер.

В таблицах переходов две первые колонки одинаковые - в них перечислены все возможные комбинации входного сигнала и состояния элемента памяти. Для того, чтобы элементарный автомат имел полную систему переходов, колонку Q(t+1) следует заполнить таким образом, чтобы во второй и третьей колонках встречались все возможные типы переходов (00, 01, 10, 11). Для триггера типа D колонка Q(t+1) и D совпадают, т.к. выходной сигнал отождествляется с состоянием, то это означает, что данный элемент является элементом задержки на 1 такт. Его характеристическое уравнение имеет вид: Q(t+1)=δ(Q(t), D(t))= DQ v D$\overline{Q}$ = D. Триггер типа Т изменяет свое состояние только при подаче на его вход «1». Это триггер со счетным входом. Его характеристическое уравнение имеет вид:

$$\mathbf{Q(t+1)=\delta(Q(t),\ T(t))=\ \overline{T}\,Q\ v\ T\overline{Q}.}$$

2.4.2 Элементы памяти с двумя информационными входами

Триггеры с двумя информационными входами имеют различное построение в зависимости от режимов использования имеющихся входов. Основными, наиболее распространенными двухвходовыми триггерами являются RS-триггер, JK-триггер, синхронизированный D-триггер. Рассмотрим подробнее каждый из них.

RS-триггер

Название этого элемента происходит от английских слов «set-reset» - «установка-сброс». Он имеет два установочных входа: S -установка в 1, R - установка в ноль (сброс). Работа описывается таблицей переходов (табл. 2.21.). На 6 и 7 наборах функция не

определена, т.к. считается, что нет необходимости устанавливать данный триггер в положение «1» и «О» одновременно. Таким образом, входная комбинация 11 для RS-триггера является запрещенной и не должна возникать в реальных условиях работы.

Характеристическое уравнение после его преобразования и минимизации имеет вид:

$Q(t+1)=\delta(Q(t), R(t), S(t))= \overline{R}\, Q \vee S.$

Это соотношение показывает, что при нулевом сигнале на входе «установка в ноль» (R=0) RS-триггер является дизъюнктором накапливающего типа. Он осуществляет логическое сложение содержимого триггера Q(t) и сигнала S(t), после чего результат операции записывается вместо первого слагаемого. В частном случае, при обнуленном триггере, осуществляется запись в триггер той информации, которая поступила на вход S. Условное графическое обозначение RS-триггера представлено на рис. 2.7.а).

Таблица 2.21.

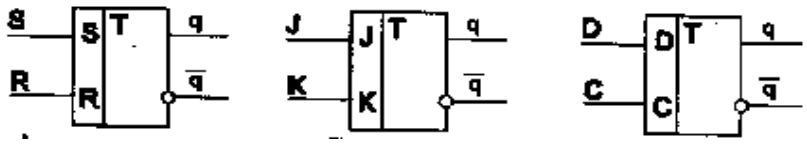
№	S	R	Q(t)	Q(t+1)
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	-
7	1	1	1	-

Таблица 2.22.

№	J	K	Q(t)	Q(t+1)
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	1
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Таблица 2.23.

№	D	C	Q(t)	Q(t+1)
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	1



а) б) в)

Рисунок 2.7- Условное графическое обозначение триггеров:
а) RS-триггер; б) JK-триггер; в) синхронизированный D-триггер.

J-K-триггер

Он имеет два установочных входа: J - установка в 1, К - установка в ноль. Работа описывается таблицей переходов (табл. 2.22.). Для него не существует запрещенных наборов входных сигналов.

Характеристическое уравнение после его преобразования и минимизации имеет вид:

$Q(t+1)=\delta(Q(t), J(t), K(t)) = \overline{K}Q \vee \overline{Q}J.$

Из этого соотношения следует, что JK-триггер является универсальным, имеющим два режима работы.

1) Режим записи и хранения информации, пришедшей по входу J, если триггер заранее был обнулен, поскольку работа обнуленного JK-триггера описывается уравнением RS-триггера. Данный режим называется RS-режимом.

2) Режим счета, который возникает при обеспечении одинаковых сигналов на обоих входах. Поскольку такой режим описывается уравнением, аналогичным уравнению Т-триггера, то его можно назвать Т-режимом. Условное графическое обозначение JK-триггера представлено на рис. 2.7.б.

D-триггер

Триггер имеет также 2 входа: информационный (D) и синхронизирующий (C). Функция на выходе триггера принимает значение информационного сигнала, если есть разрешающий сигнал по входу C (C=1). При отсутствии разрешающего сигнала (C=0), значение функции замораживается, т.е. остается равным содержимому триггера на предыдущем такте. Работа описывается таблицей переходов (табл. 2.23.).
Характеристическое уравнение триггера имеет вид:

$Q(t+1)=\delta(Q(t), D(t), C(t))= \overline{C}\, Q \vee DC.$

Из чего следует, что D-триггер является переключателем накапливающего типа: он пропускает на выход либо сигнал, приходящий по условной шине D, либо сигнал, приходящий по условной шине Q(t), в зависимости от значения управляющего сигнала C. Условное графическое обозначение триггера представлено на рис. 2.7.в.

2.4.3 Матрица переходов элемента памяти

Элемент памяти (триггер) может быть задан одним из нескольких способов: сокращенной таблицей переходов, полной таблицей переходов, характеристическим уравнением, матрицей переходов. Рассмотрим последний способ.

Для каждого из 4-х возможных переходов элементарного автомата (00, 01, 10, 11) всегда найдется значение входного сигнала, равное 0 или 1, которое вызывает данный переход. Если элементарный автомат имеет 2 или более входов, то на некоторые переходы значения входных сигналов, действующих на одном или другом входе, оказываются несущественными.

Опишем закон функционирования элементарного автомата, имеющего m входов, матрицей переходов:

	x_1	x_2	…	x_k	…	x_m
00	b_{00}^{1}	b_{00}^{2}	…	b_{00}^{k}	…	b_{00}^{m}
01	b_{01}^{1}	b_{01}^{2}	…	b_{01}^{k}	…	b_{01}^{m}
11	b_{10}^{1}	b_{10}^{2}	…	b_{10}^{k}	…	b_{10}^{m}
10	b_{11}^{1}	b_{11}^{2}	…	b_{11}^{k}	…	b_{11}^{m}

Количество строк матрицы всегда равно 4 (по количеству возможных переходов), количество столбцов равно числу входных сигналов. Элемент матрицы $\mathbf{b_{is}^{k}}$ представляет собой значение входного сигнала x_k под действием которого элементарный автомат перейдет из состояния i в состояние s. При этом b_{is}^{k} всегда равняется 0 или 1, или неопределен, если он не влияет на данный переход.

Матрица переходов элементарного автомата составляется по таблице переходов.

Рассмотрим пример построения матрицы переходов триггера.

Пусть триггер, в общем случае, задан сокращенной таблицей переходов (табл. 2.24.). Построить полную таблицу переходов триггера и матрицу переходов.

Таблица 2.24.

t		(t+1)
X	У	Q
0	0	0
0	1	Q(t)
1	0	$\overline{Q(t)}$
1	1	1

Таблица 2.25

t			t+1
X	У	Q	Q
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Таблица 2.26.

Q(t)	Q(t+1)	X	У
0	0	0	0
		0	1
0	1	1	0
		1	1
1	0	0	0
		1	0
1	1	0	1
		1	1

Полная таблица переходов триггера, построенная по сокращенной, представлена в таблице 2.25. По полной таблице переходов запишем комбинации входных сигналов, соответствующих всем возможным переходам (табл. 2.26.) Таким образом:

1. Для перехода "0-0" X=0, Y может быть равен 0 или 1
2. Для перехода "0-1" X=1, Y может быть равен 0 или 1
3. Для перехода "1-0" X может быть равен 0 или 1, а Y=0.
4. Для перехода "1-1" X может быть равен 0 или 1, а Y=1.

Тогда матрица переходов триггера запишется в виде:

0-0	0	b_1
0-1	1	b_2
1-0	b_3	0
1-1	b_4	1

где b1,b2,b3,b4 - произвольные сигналы (0 или 1).

Как правило, значение двух различных коэффициентов b_i, и b_s из одной строки матрицы являются зависимыми друг от друга и нахождение этой зависимости с ростом числа информационных входов усложняется. Установление точной взаимозависимости между входными переменными триггера является обязательным условием, обеспечивающим возможность максимального упрощения схем с памятью. В основе методики лежит таблица, в которой представлены возможные сочетания входных переменных и соответствующие им описания (табл. 2.27.).

Таблица 2.27. Таблица доопределения входных сигналов триггеров.

			Номер варианта													
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Входные сигналы		X1	0	0	1	1	00	01	01	01	01	11	001	001	011	011	0011
		X2	0	1	0	1	01	00	01	10	11	01	010	011	001	101	0101
Описание взаимозависимости	I	X1	0	0	1	1	-	b1	b1	b1	b1	-	b1	b1	b1	b1	b1
	II	X2	0	1	0	1	-	0	b1	$\overline{b1}$	1	-	$\overline{b1}\wedge b2$	$b1\vee b2$	$b1\vee b2$	$\overline{b1}\vee b2$	b2
		X1	0	0	1	1	0	-	b1	$\overline{b1}$	-	1	$\overline{b1}\wedge b2$	$b1\vee b2$	$b1\vee b2$	$\overline{b1}\vee b2$	b2
		X2	0	1	0	1	b1	-	b1	b1	-	b1	b1	b1	b1	b1	b1

С учетом доопределений входных переменных матрицы переходов некоторых стандартных триггеров имеют вид (таб. 2.28.)

Таблица 2.28. Матрицы переходов триггеров

Триггер-D			Триггер-T			Триггер-RS				Триггер-JK			
Q(t)	Q(t+1)	D	Q(t)	Q(t+1)	T	Q(t)	Q(t+1)	R	S	Q(t)	Q(t+1)	K	J
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1	0	1	1	0	1	0	1	0	1	0	1
1	0	0	1	0	1	1	0	1	0	1	0	1	0
1	1	1	1	1	0	1	1	0	0	1	1	0	0

2.5 Кодирование состояний и выходов автомата и сложность комбинационных схем

Анализ канонического метода структурного синтеза показал, что различные варианты кодирования состояний автомата приводят к различным выражениям для функций возбуждения и функций выходов.

В рассмотренном ранее примере коды состояний автомата принимали значения: a_1=00, a_2=01, a_3=11. Если принять коды: a_1=01, a_2=01, a_3=00, то таблица переходов структурного автомата примет вид (табл. 2.29).

Если в качестве запоминающего элемента применить D-триггер, то таблица формирования функций возбуждения будет совпадать с данной таблицей. Тогда функции примут вид:

Таблица 2.29

	Z_1Z_2		
$\alpha_1\alpha_2$	00	01	10
01	10	00	10
10	-	01	00
00	01	-	00

$$\varphi_1 = \bar{a}_1 a_2 \bar{z}_1 z_2 \vee \bar{a}_1 a_2 z_1 \bar{z}_2;$$

$$\varphi_2 = \bar{a}_1 \bar{a}_2 \bar{z}_1 \bar{z}_2 \vee a_1 \bar{a}_2 \bar{z}_1 z_2;$$

Если сравнить с предыдущим результатом, то нетрудно сделать вывод, что реализация этих выражений комбинационной схемой проще.

В данном случае при кодировании состояний был использован весовой метод, который может быть использован и для кодирования выходных сигналов.

Алгоритм весового метода кодирования:

1. Каждому выходному сигналу y, ставится в соответствие целое число P_i, равное числу появлений сигнала y_i в таблице выходов автомата.
2. Числа P_i сортируются по убыванию.
3. Выходной сигнал y_i с наибольшим весом (P_i max) кодируются кодом, содержащим все 0 (00...00).
4. Следующие L выходных сигналов (где L- число разрядов в двоичном векторе выходного сигнала) по списку убывания веса (см. п. 2) кодируются кодами, содержащими одну 1 (00...01, 00...10,... ,10...00).
5. Для кодирования следующих по списку убывания выходных сигналов используются все коды, содержащие две единицы, затем три единицы и т.д., пока не будут закодированы все выходные сигналы.

В результате получаем такое кодирование, при котором чем чаще встречается выходной сигнал в таблице выходов, тем меньше единиц содержится в его коде.

Кодирование внутренних состояний двоичными символами оказывает существенное влияние на стоимость комбинационной части схемы автомата. Оптимальное кодирование даст минимальную стоимость, однако алгоритм эффективного кодирования неизвестен. Тем не менее, при кодировании внутренних состояний автомата используются различные

эвристические алгоритмы, позволяющие, по крайней мере в некоторых случаях получить кодирование, близкое к оптимальному.

Д.А. Морозом предложен **эвристический алгоритм кодирования** внутренних состояний автоматов, основанный на минимизации суммарного числа изменений состояний элементов памяти автомата на всех переходах автомата.

При таком критерии уменьшается сложность схем, реализующих дизъюнкции на входах элементов памяти, т.е. минимизируется комбинационная схема автомата. Для оценки кодирования вводится коэффициент эффективности кодирования:

$K_{эф}$= W/P; где P - общее количество переходов автомата,
W - весовая функция : **$W=\Sigma t_{ij}$**

где t_{ij}- расстояние Хэмминга между кодами состояний a_i и a_j, равное числу элементов памяти, изменяющих свое состояние на данном переходе.

Отметим, что при определении весовой функции суммирование производится по всем переходам автомата. Коэффициент эффективности позволяет оценить сложность комбинационной схемы автомата: чем меньше его значение, тем меньше сложность комбинационной схемы, и оптимальный вариант - $K_{эф}=1$.

Алгоритм состоит из следующих шагов:

1. Построить матрицу M, составленную из всех пар номеров (a_r, b_r) переходов автомата.

2. Переставить строки в матрице таким образом, чтобы в каждой последующей строке содержался хотя бы один элемент из предыдущих строк.

3. Закодируем состояния из первой строки матрицы M следующим образом: Ka_1=00...00, Kb_1=00...01.

4. Вычеркнем из матрицы M первую строку с закодированными состояниями. Получим матрицу M'.

5. В начальной строке матрицы M' закодирован один элемент. Выберем из первой строки матрицы M' незакодированный элемент и обозначим его γ.

6. Построим матрицу Mγ, выбрав из матрицы M' строки, содержащие γ. Пусть Bγ=$\{\gamma_1, \gamma_2,..., \gamma_f,..., \gamma_F\}$ - множество элементов из матрицы Mγ, которые уже закодированы. Их коды обозначим через К γ_1,К γ_2,... ,К γ_f,...,К γ_F соответственно.

7. Для каждого Кγ_f (f=1, 2,...,F) найдем $C^1\gamma_f$ - множество кодов, отстоящих от Кγ_f на расстояние Хэмминга, равное 1 и еще не занятых для кодирования состояний автомата. Построим множество $D^1_\gamma = \bigcup_{f=1}^{F} C^1_{\gamma f}$.

Если $D^1\gamma$ =0, то строим новое множество $D^2_\gamma = \bigcup_{f=1}^{F} C^2_{\gamma f}$, где $C^2_{\gamma f}$ - множество кодов, у которых кодовое расстояние с кодом К γ_f равно 2. Если и $D^2\gamma$ =0, строим $D^3\gamma$ и т.д., пока не найдем $D^n_\gamma \neq 0$. Пусть $D^n_\gamma = \{K_{\delta 1},...,K_{\delta g},...,K_{\delta G}\}$.

8. Для каждого К$_{\delta g}$ находим $w_{gf}=|K_{\delta g}- K_{\gamma f}|^2$ - расстояние Хэмминга между К$_{\delta g}$ и всеми используемыми кодами $K_{\gamma f}$(f=1, 2, ...,F).

9. Находим $w_g = \sum_{f=1}^{F} w_{gf}, (g = 1,...,G).$

10. Из D^n_γ выбираем К$_\gamma$, у которого $W_g=W_{g\,min}$. Элемент y кодируем кодом К$_\gamma$.

11. Из матрицы M' вычеркиваем строки, в которых оба элемента закодированы, в результате чего получаем новую матрицу, которую также обозначаем M'. Если в матрице M' не осталось ни одной строки, переходим к п. 12, иначе к п. 5.

12. Вычисляем функцию $W = \sum t_{ij}$, где $t_{ij}=|K_j-K_j|^2$.

13. Конец.

Оценкой качества кодирования по рассмотренному алгоритму может служить число K=W/P, где P - число переходов в автомате. Очевидно, что K≥1, причем, чем меньше значение K, тем лучше результат кодирования.

Рассмотрим пример кодирования автомата, заданного таблицей переходов и выходов.

Таблица 2.30.

У	y_1	y_2	y_2	y_3
A	a_1	a_2	a_3	a_4
x_1	3	4	2	3
x_2	4	3	1	2

$$M=\begin{vmatrix}1\text{-}3\\1\text{-}4\\2\text{-}4\\2\text{-}3\\3\text{-}2\\3\text{-}1\\4\text{-}3\\4\text{-}2\end{vmatrix}\quad M'=\begin{vmatrix}\\1\text{-}4\\2\text{-}4\\2\text{-}3\\3\text{-}2\\4\text{-}3\\4\text{-}2\end{vmatrix}\quad M_\gamma=\begin{vmatrix}\\\\1\text{-}4\\2\text{-}4\\4\text{-}3\\4\text{-}2\end{vmatrix}$$

1. Строим матрицу M:

Кодируем первую строку: $K_1=00$; $K_2=01$;

2. Вычеркиваем из матрицы M 1-ю строку (1-3) и 6-ю строку (3-1). Получаем матрицу M', в первой строке которой незакодирован элемент 4. Обозначим γ=4 и запишем матрицу M_γ,. В матрице M_γ закодированы 1 и 3: $B_\gamma=\{1,3\}=\{00,01\}$ $C_{41}^1=\{10\}; C_{43}^1=\{11\}; D_4^1=\{10,11\}$.

Находим W:

$W_{10}=|00|\ |10|=3$; $W_{11}=|00|\ |11|=3$;
$|10|+|01|$ $|11|+|01|$

Выбираем $K_4=10$.

Вычеркнем из M' строку (1-4). Получим новую матрицу M', в первой строке которой не закодирован элемент 2. Обозначим g=2 и запишем матрицу M_2. В матрице M_2 закодированы элементы 3, 4:

Bγ={3,4}={01,10} $C_{23}^1=\{11\}$; $C_{24}^1=\{11\}$; $D_2^1=\{11\}$.

$W_{11}=\begin{vmatrix}11\\10\end{vmatrix}+\begin{vmatrix}11\\01\end{vmatrix}+\begin{vmatrix}01\\11\end{vmatrix}+\begin{vmatrix}10\\11\end{vmatrix}=4$. В связи с отсутствием других вариантов кодов выбираем $K_2=11$. Таким образом: K1=00; K2=11; K3=01; K4=10.

Вычислим функцию $W=\begin{vmatrix}00\\01\end{vmatrix}+\begin{vmatrix}00\\10\end{vmatrix}+\begin{vmatrix}11\\10\end{vmatrix}+\begin{vmatrix}11\\01\end{vmatrix}+\begin{vmatrix}01\\11\end{vmatrix}+\begin{vmatrix}01\\00\end{vmatrix}+\begin{vmatrix}10\\01\end{vmatrix}+\begin{vmatrix}10\\11\end{vmatrix}=9$.

Вычислим K=W/P=9/8=1,125.

2.6 Обеспечение устойчивости функционирования цифровых автоматов. Гонки в автоматах

Одна из главных задач, решаемых на этапе структурного синтеза синхронных цифровых автоматов с памятью, заключается в обеспечении устойчивости их функционирования. Понятие устойчивости связано с разработкой такой принципиальной электрической схемы автомата, которая обеспечивала бы его функционирование в соответствии с таблицей переходов и выходов автомата.

Неправильное функционирование автомата (неустойчивая его работа) связано с особенностями физической реализации логических элементов и элементов памяти его схемы, а также различными величинами задержек распространения сигнала в элементах и комбинационных схемах.

Рассмотрим процесс обеспечения устойчивости функционирования автомата более подробно. После поступления очередного входного сигнала и формирования сигналов возбуждения на входах элементов памяти автомат переходит в новое состояние. При этом происходит формирование новых сигналов возбуждения по цепям обратных связей (с выходов элементов памяти через логические элементы на входы элементов памяти), и автомат переходит в новое состояние и т. д.

Таким образом, автомат, в общем случае, не может остановиться в каком-то определенном состоянии и начинает функционировать в режиме *генератора состояний.* Для устранения такого эффекта используют синхросерию — последовательность специальных (обычно прямоугольных) сигналов, подаваемых на входы элементов памяти и разрешающих поступление очередных сигналов возбуждения на входы элементов памяти только с приходом очередного синхросигнала. При отсутствии синхросигнала сигнал возбуждения не поступает на вход элемента памяти, и элемент памяти цифрового автомата не переключается, т. е. остается в каком-то состоянии.

Практически подключение синхросерии осуществляется к специальным входам элемента памяти, обозначаемым символом С на рисунках и называемым синхровходами. Введение синхросерии, однако, не обеспечит устойчивого функционирования автомата, если не учитывать некоторые особенности. Если при переходе из одного состояния в другое должны изменять свои состояния сразу несколько элементов памяти, то между ними начинаются **состязания.** Тот элемент, который выиграет состязания, по цепям обратной связи может изменить сигналы на входах других элементов памяти до того, как они изменят свои состояния. Это может привести к переходу автомата в состояние, не предусмотренное графом.

Если в процессе перехода из состояния a_m в состояние a_s под действием входного сигнала x_i автомат может оказаться в некотором промежуточном состоянии (a_i, a_k) в зависимости от того, какой из элементов памяти выиграл состязания, но, затем, при этом же входном сигнале перейдет в состояние a_s, то такие состязания называются допустимыми, или не критическими. Но если произойдет переход в состояние a_k (не предусмотренное графом переходов автомата) и правильность работы автомата нарушается, то такие состязания называются недопустимыми (критическими) или **гонками.** Пусть автомат должен выполнить переходы, изображенные на рис. 2.8.

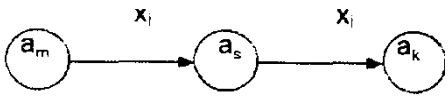


Рисунок 2.8

Возможны следующие ситуации:

а) если очередной синхросигнал на входы элементов памяти автомата поступает раньше, чем кончились переходные процессы в его комбинационной схеме возбуждения и элементах памяти после поступления входного сигнала x_i, то возможно неправильное формирование сигнала возбуждения на входе одного или нескольких элементов памяти автомата, т. е. автомат может вместо перехода из состояния a_m в состояние a_s по входному сигналу x_i осуществить ложный переход в некоторое состояние a_t по тому же самому входному сигналу x_i;

б) если длительность входного сигнала x_i превышает длительность перехода автомата из состояния a_m в состояние a_s, то автомат может (при поступлении очередного синхросигнала) проскочить состояние a_s и попасть сразу в состояние a_k за счет двойного срабатывания автомата по входному сигналу x_i. Иными словами, состояние a_s может оказаться неустойчивым.

2.6.1 Методы устранения гонок в автоматах

Для обеспечения устойчивого функционирования автомата нужно разнести во времени момент подачи информации на входы его элементов памяти и момент снятия информации с выходов элементов памяти. При таком разнесении формирование очередного сигнала возбуждения любого элемента памяти в момент появления синхросигнала осуществляется только по значениям состояний элементов памяти в предшествующий момент времени, а переходные процессы в элементах памяти не влияют на формирование сигнала возбуждения (выходы элементов памяти отключены).

Естественно, что период следования синхросигналов при этом должен выбираться исходя из учета окончания переходных процессов, связанных с задержками распространения входного для автомата сигнала по логическим элементам комбинационной схемы возбуждения. В результате устойчивость функционирования цифрового автомата может быть обеспечена, например, использованием двухэтажной памяти. В этом случае каждый элемент памяти дублируется, и перепись информации из нижнего элемента памяти в верхний осуществляется по отсутствию синхросигнала (рис. 2.9.).

Сигналы обратных связей, используемые для формирования функций возбуждения, и сигналы выходов автомата снимаются с выходов элемента памяти верхнего яруса. При такой организации памяти автомата отсутствует опасность формирования повторного сигнала возбуждения по одному и тому же синхросигналу и перехода автомата в новое состояние. Последнее связано с тем, что переход в рабочее состояние автомата завершается после окончания действия синхросигнала.

Однако использование двойной памяти автомата приводит к замедлению работы автомата. Если обычно период синхросигналов выбирается из расчета, что сигнал возбуждения элемента памяти успеет пройти по самой длинной цепочке логических элементов и переключить элемент памяти, то здесь период нужно удлинить по крайней мере на 3t где t- задержка распространения сигнала в логическом элементе (1t- на инвертор и 2t - на второй элемент памяти).

В случаях, когда из соображений быстродействия двухэтажную память использовать нельзя, прибегают к многофазной системе тактирования входных сигналов автомата. Так, для случая двухфазной синхронизации синхросериями СС1 и СС2 вместо одного входного сигнала x_i, (рис. 2.8.) используются два разных: $x_i \bullet CC1$ и $x_i \bullet CC2$ (рис. 2.10.)

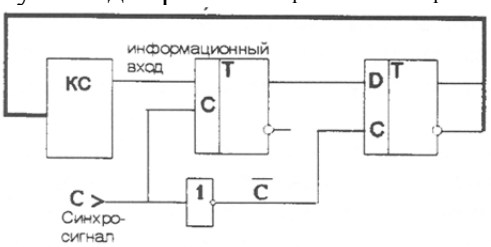


Рисунок 2.9

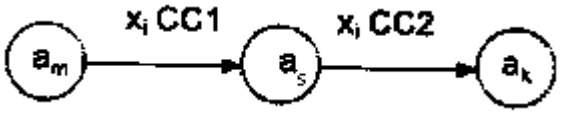


Рисунок 2.10

Таким способом устойчивость функционирования автомата обеспечивается автоматически. При двухфазной синхронизации необходимо, чтобы все дуги графа переходов автомата можно было бы разметить символами СС1 иСС2 так, что для любой вершины графа все выходящие из нее дуги отмечались бы символом одной синхросерии (например, СС1, а все дуги, заходящие в ту же вершину графа переходов автомата,— символом другой синхросерии (например, СС2). Если граф переходов автомата содержит контур нечетной длины, то такая разметка невозможна.

Однако ее можно сделать, преобразовав контур нечетной длины в контур четной длины, добавив дополнительную вершину или состояние автомата с пустым выходным сигналом. Задача преобразования произвольного графа с нечетными контурами к графу с четными контурами решается методами теории графов, в частности с использованием понятия цикпоматического числа графа и метода построения матрицы фундаментальных циклов графа.

Кроме описанных выше случаев, устойчивость функционирования цифрового автомата с памятью может быть частично обеспечена с помощью специальных мер, принятых относительно устранения в схеме автомата эффекта гонок. Это связано с тем, что элементы памяти имеют различные времена срабатывания. Различны также задержки сигналов возбуждения, поступающих на входы элементов памяти по цепочкам логических элементов различной длины.

Если при переходе автомата из одного состояния в другое, должны переключиться сразу несколько элементов памяти, то между ними начинаются гонки, (состязания), что может привести к неправильной работе автомата. В самом деле, при переходе автомата из состояния a_i; в состояние a_j на некоторый, хотя и очень короткий, промежуток времени может возникнуть промежуточное состояние автомата, отличное от a_i и a_j. Например, при переходе из a_i (0110) в a_j (1010) изменяют свое состояние два первых элемента памяти. Из-за состязаний может возникнуть состояние 1110 (или 0010), которое может привести к изменению состояния третьего или четвертого элемента памяти.

В последнем случае после окончания переходных процессов автомат уже не попадает в состояние a_j. При использовании двухэтажной памяти гонки в автомате не возникают, так как изменение состояния автомата происходит в то время, когда синхросигнал отсутствует.

Следует заметить, что современная элементная база включает в себя элементы памяти с уже встроенной двухэтажной памятью (например, JK-триггер). Существует еще один способ устранения гонок в автоматах, связанный со специальным кодированием состояний автомата, которое называется противогоночным кодированием.

Частным случаем противогоночного кодирования является соседнее кодирование, при котором состояния автомата, связанные дугой на графе переходов, кодируются двоичными векторами, отличающимися друг от друга только в одном разряде. Для проведения соседнего кодирования в графе переходов автомата не должно быть контуров нечетной длины. Примеры графов переходов автоматов, состояния которых закодированы соседним образом, представлены на рис.

Отметим, что проблема состязаний и некоторые другие вопросы обеспечения устойчивости работы автоматов возникли лишь с появлением потенциальной элементной базы. Используемая ранее (в ЭВМ второго поколения) импульсно-потенциальная

элементная база предусматривала применение статических триггеров со встроенной задержкой.

При этом величина задержки выбиралась большей длительности импульсного сигнала, поступающего на вход триггера. Тем самым переходные процессы формирования сигналов на выходах элементов памяти автомата начинались лишь после окончания входного импульсного сигнала и, следовательно, не оказывали воздействие на входы этих же элементов памяти по цепям обратной связи. Нечто подобное осуществляется теперь в потенциальной элементной базе введением не совпадающих во времени серий синхронизирующих сигналов, в особенности при организации двухэтажной памяти.

3 МИКРОПРОГРАММНЫЕ АВТОМАТЫ

3.1 Основные понятия. Принцип микропрограммного управления

Рассмотренные методы и приемы синтеза дискретных устройств (ДУ) использовали их представление в виде совокупности двух основных блоков: комбинационного логического и блока элементов памяти. Такой подход обладает универсальностью и обеспечивает хорошие результаты при построении относительно несложных ДУ. Однако полученные на его основе процедуры синтеза ДУ оказываются чрезмерно громоздкими и трудоемкими при построении устройств средней и большой сложности, имеющих важное практическое значение. Работа таких устройств обычно заключается в реализации некоторого алгоритма обработки информации, т. е. в выполнении упорядоченной последовательности определенных операций над поступающими данными. При построении таких ДУ целесообразно использовать принцип микропрограммного управления, состоящий в следующем:

1) любая операция, реализуемая устройством, рассматривается как сложное действие, которое разделяется на последовательность элементарных действий, называемых микрооперациями;

2) для управления порядком следования микроопераций используются логические условия x_i, принимающие в зависимости от результатов выполнения микроопераций значения 1 или 0;

3) процесс выполнения операций в устройстве описывается в форме алгоритма, представленного в терминах микроопераций и логических условий и называемого микропрограммой;

4) микропрограмма используется как форма представления функции устройства, на основе которой определяются его структура и порядок функционирования.

Все сказанное можно рассматривать, как содержательное описание принципа микропрограммного управления.

3.2 Концепция управляющего и операционного автоматов

При использовании описанного принципа принято делить ДУ на две части: операционный автомат (ОА) и управляющий автомат (УА) (рис. 3.1).

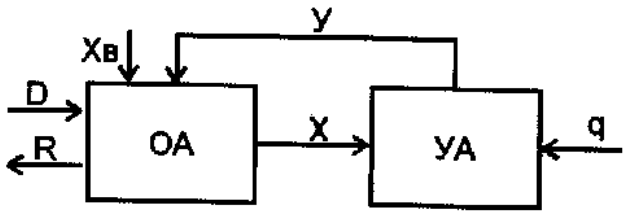


Рисунок 3.1-Обобщенная структурная схема микропрограммного дискретного устройства

ОА предназначен для хранения поступающей информации D, выдачи результатов выполнения операций R, выполнения заданного набора микроопераций, выработки значений логических условий $X=(x_0,x_1,..., x_m)$, которые являются оповещающими сигналами для управляющего автомата. УА генерирует последовательность управляющих сигналов $Y=(y_1,y_2,...,y_n)$ в соответствии с заданной микропрограммой и со значениями логических условий X. Каждый управляющий сигнал инициирует выполнение соответствующей микрооперации в ОА.

В общем случае ДУ предназначается для выполнения ряда микропрограмм, и на УА подается внешний сигнал q, в соответствии с которым начинается выполнение той или иной микропрограммы. Если ДУ является частью системы обработки информации, то оно может также обмениваться специальными сигналами логических условий X_B и управления Y_B с другими блоками системы.

В состав ОА входят главным образом типовые функциональные узлы: регистры, счетчики, сумматоры, дешифраторы, шифраторы, арифметико-логические устройства (АЛУ), схемы сравнения, блоки памяти, схемы пересылки данных и т. п. Число элементов памяти (ЭП), содержащихся в ОА, определяется разрядностью обрабатываемых данных $n_Д$, которая может быть достаточно большой. Однако трудоемкость и сложность проектирования ОА, как правило, слабо зависят от $n_Д$ в силу широкого использования стандартных узлов. Таким образом, ОА является *исполнительной частью* устройства; его состав и структура могут быть одинаковыми для реализации многих алгоритмов одного класса.

Элементарный неделимый акт обработки информации в операционном автомате, происходящий в течение одного момента автоматного времени (одного такта работы автомата), называется ***микрооперацией.*** Примерами микроопераций могут служить «Сдвиг информации», «+1», «Инверсия переменной» и т. д.

Если в операционном автомате одновременно реализуется несколько микроопераций, то такое множество микроопераций называется ***микрокомандой.*** Не исключен случай, когда множество микроопераций, образующих микрокоманду, пусто. Реализация такой микрокоманды в операционном автомате равносильна отсутствию выполнения каких-либо элементарных операций. В случае синхронных дискретных устройств пустая микрокоманда интерпретируется как пропуск такта, когда никакие сигналы от управляющего автомата на операционный автомат не поступают.

Микрооперации возбуждаются выходными сигналами управляющего автомата, а их последовательность во времени определяется функциями перехода управляющего автомата.

Совокупность микрокоманд и функций перехода образует ***микропрограмму.*** Таким образом, для описания микропрограммы необходимо задать множество микрокоманд и функций перехода, определяющих порядок их выполнения. Для описания микропрограмм удобно использовать язык граф-схем алгоритмов (ГСА).

3.3 Управляющие автоматы с жесткой и программируемой логикой

Объем оборудования УА зависит от сложности реализуемого алгоритма и от структуры этого автомата, которую можно выполнить в трех вариантах.

1. УА с жесткой (схемной, произвольной) логикой, при которой переключательные функции, необходимые для формирования заданной последовательности управляющих сигналов У, реализуются с помощью логических элементов с произвольными связями (обычно с применением схем с малой и средней степенями интеграции). Здесь используется аппаратный подход к реализации устройства.

2. УА с хранимой в памяти (гибкой, программной) логикой, при которой сигналы У вырабатываются на основе совокупности управляющих слов, хранимых в памяти автомата. В этом случае составленные микропрограммы используются в явной форме и обычно записываются в постоянные запоминающие устройства (ПЗУ), выполненные на основе полупроводниковых БИС большой емкости, что позволяет обеспечить регулярность структуры УА и его компактность; здесь используется аппаратно-программный подход к реализации устройства.

3. УА на основе программируемых логических матриц (ПЛМ), в которых заданные функции реализуются с помощью БИС ПЛМ, что позволяет сочетать многие достоинства первых двух вариантов.

Таким образом, использование принципа микропрограммного управления позволяет упорядочить и упростить процедуру логического проектирования ДУ, обеспечить регулярность их структуры, а также открывает возможность широкого применения современных БИС. Принцип микропрограммирования применяется при создании микропроцессоров и устройств на их основе. Это не только позволяет упорядочить управление, но и дает возможность формировать систему команд микропроцессоров по своему усмотрению, исходя из имеющейся системы микрокоманд.

Рассмотрим порядок проектирования микропрограммного ДУ, который состоит из следующих основных этапов :

Запись алгоритма. По описанию отдельных алгоритмов, реализуемых устройством, составляется их формализованная запись в виде граф-схем алгоритмов (ГСА). Для этого составляется список необходимых микроопераций $У_j$, и соответствующих им управляющих сигналов y_j, а также логических условий x_i;. Далее при необходимости производится минимизация числа вершин ГСА и составляется объединенный ГСА, являющийся формой здания ДУ для выполнения следующих этапов.

Построение ОА. В общем случае ОА может быть построен по канонической схеме автомата и содержит три основные части: блок элементов памяти для хранения операндов, а также промежуточных и конечных результатов; комбинационную схему, реализующую набор микроопераций; комбинационную схему, вырабатывающую значения логических условий. Как уже отмечалось, при построении ОА целесообразно применять типовые узлы, а также стремиться использовать отдельные узлы для выполнения нескольких микроопераций.

Построение УА. Сначала выбирают вариант структуры УА, учитывая требования быстродействия, допустимый объем аппаратуры и другие ограничения. Далее осуществляется синтез УА в соответствии с процедурой, зависящей от принятой структуры автомата.

В результате выполнения этих этапов составляют структурные схемы ОА и УА и переходят к техническому проектированию, которое включает вопросы практической реализации схемы устройства на выбранной элементной базе, введение необходимых развязывающих, усиливающих и формирующих каскадов, компоновку деталей на платах, составление монтажных схем и выдачу технической документации.

3.4 Граф схемы алгоритмов

ГСА - это ориентированный связный граф, задающий последовательность выполнения операций данного алгоритма и содержащий ряд операторных и условных вершин, а также одну начальную и одну конечную вершины. Операторной называется вершина, - которой сопоставляется одна или несколько микроопераций и отмечается соответствующими управляющими сигналами У, а условной - вершина, которой сопоставляется некоторое логическое условие *X.*

Любая вершина ГСА, кроме вершины «Начало», имеет по одному входу. Вершина «Начало» входов не имеет. Вершина «Начало» и любая операторная вершина имеют по одному выходу. Вершина «Конец» выходов не имеет. Любая условная вершина имеет два выхода, помечаемых символами «Да» и «Нет»: Вместо этих символов могут быть использованы цифры «1» и «О» соответственно. Изображение вершин «Начало», «Конец», операторной вершины и условной вершины ГСА представлено на рис. 3.2.

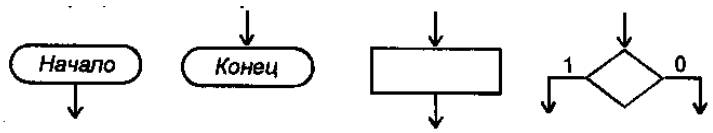


Рисунок 3.2-Графы схемы алгоритмов

ГСА составляют так, чтобы обеспечить выполнение необходимых операций и проверку логических условий в соответствии со словесным описанием алгоритма.

На основании перечня микроопераций и реализующих их функциональных узлов составляется структурная схема ОА. Здесь широкими стрелками показаны шины, по которым передается информация, а тонкими - сигналы у, управляющие работой отдельных узлов или передачей информации по шинам.

ГСА должна удовлетворять следующим условиям:

1. Входы и выходы вершин соединяются друг с другом с помощью направленных всегда от выхода к входу.
2. Каждый выход соединен только с одним входом.
3. Любой вход соединяется, по крайней мере, с одним выходом.
4. Любая вершина ГСА лежит, по крайней мере, на одном пути из вершины «Начало» в вершину «Конец».
5. Один из выходов условной вершины может соединяться с ее входом, что недопустимо для операторной вершины. Такие условные вершины иногда называются возвратными.
6. В каждой условной вершине записывается логическое условие из множества логических условий. Разрешается в различных условных вершинах записывать одинаковые логические условия.
7. В каждой операторной вершине записывается оператор, представляющий собой выходной сигнал или совокупность выходных сигналов управляющего автомата. Разрешается в различных операторных вершинах записывать одинаковые операторы.

3.5 Содержательные граф-схемы алгоритмов

Обычно при проектировании различных устройств предварительно составляется так называемая содержательная ГСА, в которой внутри условных и операторных вершин записаны логические условия и микрооперации в содержательных терминах.

В качестве примера построим содержательную ГСА устройства, вычисляющего функцию знака числа:

$$S = \begin{cases} -1, & x < 0 \\ 0, & x = 0 \\ 1, & x > 0 \end{cases}$$

Соответствующая содержательная ГСА представлена на рис. 3.3.

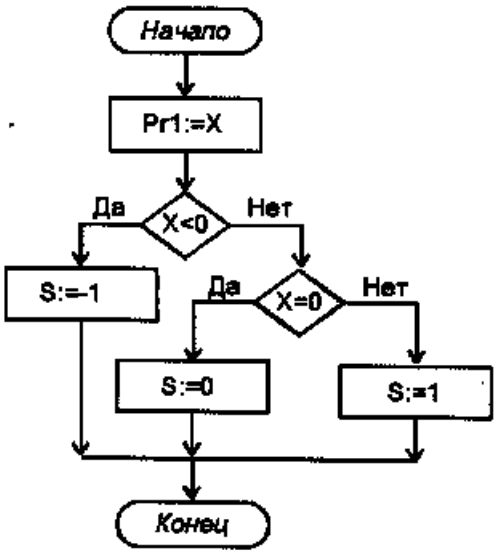


Рисунок 3.3- Содержательная ГСА функции определения знака числа

3.6 Синтез управляющего автомата по граф-схеме алгоритма

Конечный управляющий автомат, реализующий микропрограмму работы дискретного устройства, принято называть микропрограммным автоматом. Как уже отмечалось, микропрограмма отображается с помощью ГСА. Рассмотрим последовательность этапов синтеза управляющего автомата по его ГСА.

1. Запись словесного алгоритма функционирования операционного автомата (выполняемых операций) с учетом структуры операционного автомата.
2. Построение содержательной ГСА функционирования операционного автомата.
3. Построение отмеченной ГСА с учетом типа автомата.
4. Построение графа переходов автомата или таблицы переходов.
5. Проведение структурного синтеза автомата по его графу переходов известными методами, например, с помощью канонического метода структурного синтеза.

Построение отмеченной ГСА производится по содержательной ГСА. Для автоматов Мили и Мура процедура разметки имеет различия.

3.6.1 Построение отмеченной ГСА автомата Мили

Если необходимо построить микропрограммный автомат Мили, то содержательная ГСА управляющего автомата размечается в соответствии со следующими правилами:

1) символом состояния a_1 отметить вход вершины, следующей за вершиной «Начало», а также вход вершины «Конец»;

2) входы всех вершин, следующих за операторными, должны быть отмечены символами а с последовательными индексами;

3) если выход вершины отмечается, то только одним символом;

4) входы различных вершин, за исключением вершины «Конец», отмечаются различными символами;

5) содержательные термины микроопераций и логических условий. заменяются их условными обозначениями: в каждой операторной вершине последовательно проставляются символы выходных сигналов, если в различных операторных вершинах записаны одинаковые микрооперации, то разрешается их отмечать одинаковыми символами выходных сигналов; если в различных условных вершинах записаны одинаковые логические условия, то разрешается их отмечать одинаковыми символами входных сигналов.

Отмеченная ГСА для содержательной ГСА (рис. 3.3.) после разметки по приведенному алгоритму представлена на рис. 3.4.

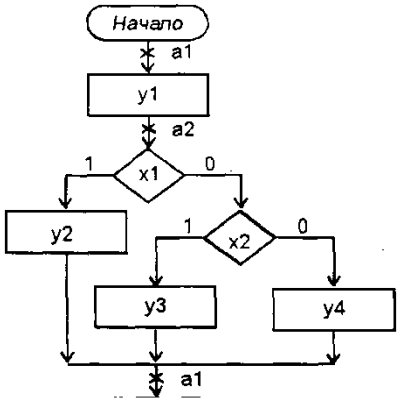


Рисунок 3.4- Отмеченная ГСА автомата Мили

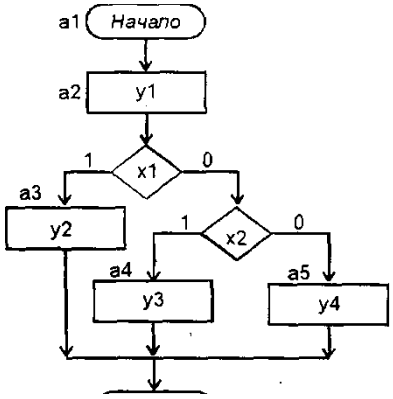


Рисунок 3.5-Отмеченная ГСА автомата Мура

3.6.2 Построение отмеченной ГСА автомата Мура

Если необходимо построить микропрограммный автомат Мура, то содержательная ГСА управляющего автомата размечается в соответствии со следующими правилами:

1) символом a_1 отмечаются вершины «Начало» и «Конец»;

2) различные операторные вершины отмечаются различными символами;

3) все операторные вершины должны быть отмечены.

4) содержательные термины микроопераций и логических условий. заменяются их условными обозначениями.

Содержательная ГСА (рис. 3.3.) после разметки по приведенному алгоритму представлена на рис. 3.5.

После получения отмеченной ГСА строится граф переходов автомата. Он имеет столько различных вершин, сколько различных букв a_i с индексами имеется в отмеченной ГСА. Каждая вершина графа переходов автомата отмечается буквой а с соответствующим индексом.

Между двумя вершинами графа имеется дуга, если на отмеченной ГСА между вершинами с метками a_i и a_k, имеется путь. Над дугой ставится входной сигнал, равный конъюнкции логических условий соответствующего пути в отмеченной ГСА. При этом

выполнению логического условия соответствует переменная без отрицания, а невыполнению логического условия - переменная с отрицанием на соответствующей дуге графа переходов автомата.

Если в отмеченной ГСА между упомянутыми вершинами с метками a_i и a_k имеется несколько путей, то в графе переходов автомата на дуге, связывающей a_i и a_k через символ дизъюнкции перечисляются все конъюнкции, соответствующие имеющимся путям.

Если строится граф переходов автомата Мура, то символы микроопераций (выходные сигналы управляющего автомата) записываются около соответствующих его вершин. Для автомата Мили символы микроопераций записываются на соответствующих дугах при конъюнкциях логических условий, описывающих путь через операторную вершину с рассматриваемой микрооперацией.

Если в отмеченной ГСА имеется безусловный переход между операторными вершинами, т. е. путь, не проходящий ни через какие условные вершины, то на графе переходов автомата ему соответствует дуга, которой приписывается входной сигнал «1», показывающий, что данный переход в автомате осуществляется при поступлении очередного синхросигнала.

В дальнейшем синтез проводится с помощью описанного ранее метода структурного синтеза. Подчеркнем, что входными сигналами синтезируемого структурного автомата являются конъюнкции булевых переменных (или дизъюнкции конъюнкций), каждая из которых отображает путь через соответствующие условные вершины отмеченной ГСА, а выходными сигналами — микрооперации, обозначающие либо вершины, либо дуги графа переходов автомата, в зависимости от его типа. Используя канонический метод структурного синтеза, можно построить функциональную схему автомата.

3.7 Построение УА с программируемой логикой на основе ПЗУ

В отличие от УА с жесткой логикой, закон функционирования которого обеспечивается определенным образом соединенными логическими элементами, в автоматах, построенных на основе ПЗУ, заданная микропрограмма реализуется в явной форме и хранится в памяти в виде последовательности управляющих слов. Управляющее слово определяет порядок работы устройства в течение одного такта и называется микрокомандой (МК). Она содержит информацию о микрооперациях, которые должны выполняться в данном такте, и (или) об адресе следующей микрокоманды.

Формат МК в общем случае может иметь операционно-адресную структуру.

Операционная	Адресная

Микрокоманда может содержать следующие части:

У		X		A		P	
1	m	1	L	1	P	1	k

- операционную часть Y, состоящую из одного или нескольких полей, в каждом записывается номер выходного сигнала y_j, вырабатываемого в данном такте;

- адресную часть, состоящую из поля X, в которое записывается номер логического условия X_i (обычно единственного), проверяемого в данном такте;

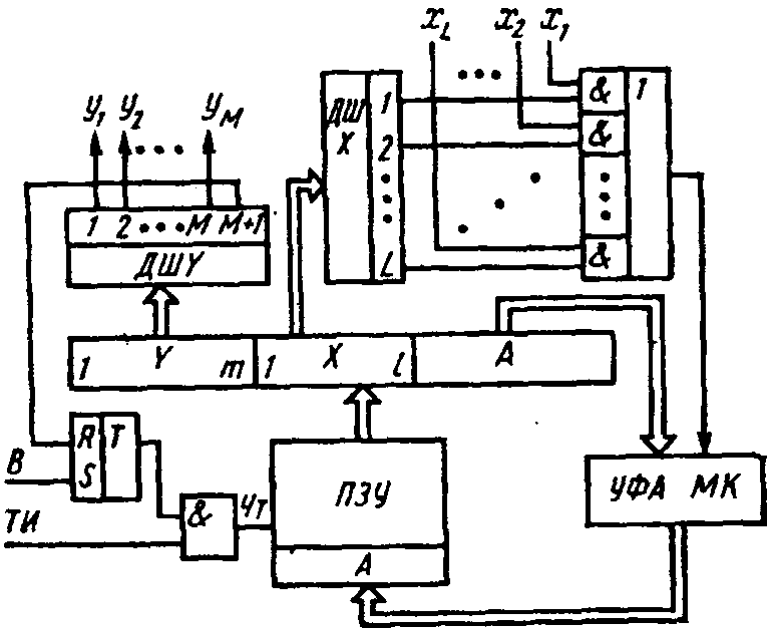


Рисунок 3.6- Обобщенная структурная схема управляющего автомата, построенного на основе ПЗУ

- а также из поля А, *в* которое записывается информация об адресе следующей МК;
- служебную часть Р, содержащую вспомогательную управляющую информацию.

Обобщенная структурная схема УА, выполненного на основе ПЗУ, дана на рис. 3.6.

Перед началом работы на УА подается сигнал СБРОС, устанавливающий все триггеры автомата и регистра микрокоманд (РМК) в нулевое состояние. Этим обеспечивается занесение содержимого нулевой ячейки ПЗУ в РМК при поступлении первого тактового импульса после подачи стартового сигнала В.

С помощью дешифратора ДШУ вырабатывается соответствующий выходной сигнал y_j, а с помощью ДШХ определяется номер логического условия X_i, проверяемого в данном такте. В зависимости от значения X_i, прошедшего через схему выбора ЛУ, и информации, поступающей из адресного поля А, устройство формирования адреса следующей МК (УФ АМК) вырабатывает адрес ячейки ПЗУ, содержимое которой будет переписано в РМК в следующем такте. На УФ АМК может также поступать внешний управляющий сигнал V, обеспечивающий, например, выбор определенного алгоритма из тех, чьи микропрограммы хранятся в ПЗУ автомата.

Схема управления СУ (в некоторых вариантах УА она может отсутствовать) разрешает работу ДШУ или ДШХ в зависимости от содержимого служебной части Р формата команд. В последнем такте выполнения микропрограммы на выходе ДШК вырабатывается дополнительный сигнал $y_{м+1}$, используемый как сигнал F, останавливающий работу автомата и осуществляющий сброс всех его триггеров.

Таким образом, структура УА с хранимой в ПЗУ логикой стандартна, и в этом заключается одно из преимуществ рассматриваемой реализации автомата. Поэтому основные усилия направляются не на получение структурной схемы, а на составление кодированной микропрограммы, которая записывается в ячейки ПЗУ, т. е. центр тяжести при разработке УА смещается с аппаратных на программные средства.

Процедура построения УА с хранимой логикой по имеющейся ГСА заключается в следующем.

1. Выбирают способ адресации и формат микрокоманд, причем стремятся сократить число двоичных разрядов в формате МК, что, как правило, позволяет уменьшить объем

оборудования ПЗУ. При этом учитывают реальное быстродействие отдельных узлов УА и необходимость обеспечения заданного быстродействия автомата в целом. При необходимости используют структурные методы повышения быстродействия УА.

2. Производят разметку ГСА в соответствии с правилами, которые определяются выбранным способом адресации.

3. Составляют кодированную микропрограмму в виде таблицы, строки которой соответствуют отметкам на ГСА.

4. Выбирают типы необходимых микросхем и составляют структурную и принципиальную схемы автомата.

Выполнив перечисленные этапы, переходят к технической реализации УА, которая во многом зависит от способа записи информации в используемое ПЗУ.

Рассмотрим особенности выполнения отдельных этапов указанной процедуры.

При построении УА используются главным образом два вида адресации: а) принудительная (в каждой МК указывается адрес следующей МК);

б) естественная (адрес следующей МК в явном виде указывается лишь в некоторых МК, а в остальных случаях он принимается равным увеличенному на единицу адресу предыдущей микрокоманды).

Формат МК при принудительной адресации может содержать как два адресных поля АО, А1, (рис. 3.7. а) так и одно АО (рис. 3.7.б).

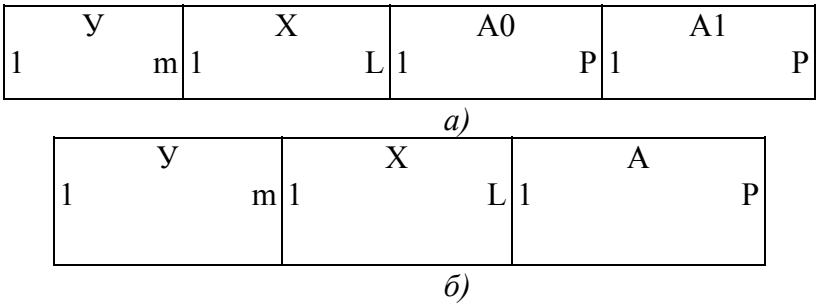


Рисунок 3.7

В первом случае адрес следующей МК определяется в зависимости от значения проверяемого в данном такте условия Xi следующим образом: в качестве адреса используется содержимое поля *АО,* если Xi =0*,* и поля А1 - если Xi = 1; безусловные переходы осуществляются по адресу *АО.*

Во втором случае переходы при xi = 0, а также безусловные переходы осуществляются по адресу *АО, а* переходы при Xi = 1 осуществляются к ячейке ПЗУ с адресом А1= *АО* + 1. Добавление единицы к *АО* может быть осуществлено с помощью комбинационной схемы инкрементора в блоке УФАМК.

При использовании двух адресных полей *АО* и А1 разметка ГСА осуществляется следующим образом.

1. Начальная вершина отмечается символом s_0.

2. Каждая операторная вершина, а также конечная вершина отмечаются символом Si, отличным от других вершин. Если число выходных сигналов y_j, записанных в некоторой вершине, превышает число операционных полей в формате команды, то число отметок у такой вершины увеличивают соответствующим образом.

3. Отмечается также каждая условная вершина, если ее вход связан с входом другой условной вершины; это вызвано тем, что в каждом такте анализируется только одно логическое условие *xi.*

Далее каждой отметке si сопоставляется ячейка ПЗУ с тем же адресом (номером) и таким образом составляется таблица содержимого ПЗУ. Эта таблица является основным результатом логического проектирования автомата наряду с принципиальной схемой УА.

Разметка ГСА при использовании единственного адресного поля *АО* осуществляется по этим же правилам, к которым добавляется еще одно:

4. Присваиваются дополнительные отметки s', и s", и т.д. каждой условной вершине, к которой подходит несколько стрелок от других условных вершин, так чтобы общее число отметок у такой вершины было равно числу упомянутых стрелок.

Необходимость увеличения числа отметок и числа используемых ячеек ПЗУ обусловлена ограничениями в расположении микрокоманд в ячейках ПЗУ из-за взаимной связи адресов *АО* и А1 = А0 + 1. Быстродействие УА несколько снижается по сравнению со случаем использования двух адресных полей за счет расхода времени на работу инкрементора. Однако исключение поля Л, из формата МК позволяет уменьшить разрядность ПЗУ.

Дальнейшее сокращение разрядности ПЗУ достигается путем перехода к естественной адресации микрокоманд, при которой обычно используются МК двух типов: операционные и управляющие. (рис. 3.8., а, *б).* Типы МК различаются по значению одноразрядного поля признака Р: 0, если МК операционная и 1, если МК управляющая

0	Y1	Y2

а) операционная микрокоманда

1	X	A

б) управляющая микрокоманда

Рисунок 3.8

Вычисление адреса следующей МК производится с помощью счетчика микрокоманд (СМК), который предусматривается в структурной схеме УА (рис. 3.9.). Операционная МК задает коды вырабатываемых сигналов yj и после ее выполнения автомат переходит к следующей МК по порядку их расположения в ячейках ПЗУ, т. е. осуществляет переход по адресу (СМК)+1, где СМК обозначает содержимое счетчика микрокоманд.

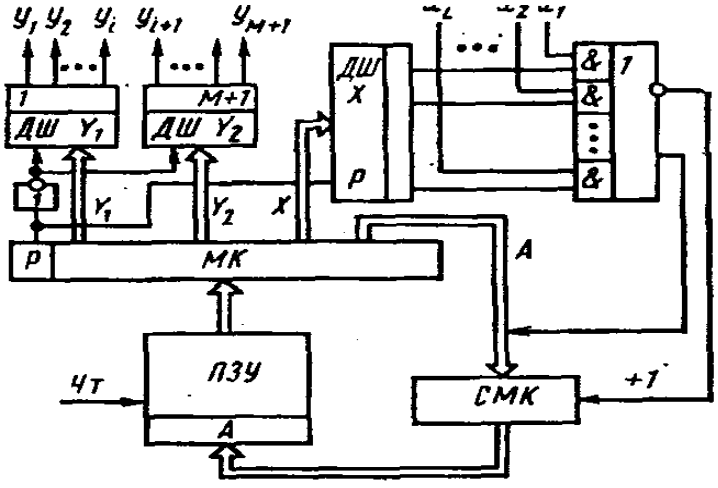


Рисунок 3.9-Структурная схема УА на основе ПЗУ при использовании естественной адресации

Управляющая МК, содержащая поле логического условия *X* и адресное поле *A,* используется для изменения естественного порядка выполнения МК, т. е. для осуществления условных и безусловных переходов в соответствии со значением проверяемого условия *Xi.* Если Xi=1, то переход осуществляется по адресу, записанному в поле *A,* для чего его содержимое переписывается в СМК. Если *Xi* = 0 или осуществляется безусловный переход, то следующую МК выбирают по адресу (СМК)+1. Таким образом, каждый такт работы УА разделяется на ряд микротактов, в течении которых выполняются действия по формированию выходных сигналов yj и выработке адреса следующей МК.

Разметку ГСА осуществляют по правилам, в которых учитывается то обстоятельство, что анализ *Xi* и выработка сигналов *yj* происходят теперь в разных тактах: начальная вершина отмечается символом s_0, каждая из остальных вершин получает отличную от других отметку si; каждая вершина, к которой подходит *h* стрелок, получает дополнительные отметки s_i^j (1=1,2, ..., h-1) так, чтобы общее число отметок у этой вершины стало равным *h.*

При составлении микропрограммы основной отметке каждой операционной вершины ставится в соответствие операционная МК, а основной отметке каждой условной вершины - управляющая МК, реализующая условный переход. Каждой дополнительной отметке s_i^j ставится в соответствие управляющая МК, реализующая безусловный переход в ячейку ПЗУ, соответствующую основной отметке si.

Составление таблицы начинают с отметки s1 и последовательно рассматривают вершины ГСА в направлении стрелок. При проходе через условную вершину сначала двигаются по направлению стрелок, отмеченных нулем (поскольку такому движению соответствует естественная адресация МК). Адресные поля управляющих МК временно остаются незаполненными. Дойдя до конечной отметки, возвращаются вверх по таблице до первой управляющей МК незаполненным адресным полем и записывают в это поле адрес следующей по порядку свободной ячейки. Далее продолжают движение по ГСА от условной вершины, которой соответствует данная управляющая МК, в направлении дуги, отмеченной единицей. Описанную процедуру возвращения вверх по таблице повторяют до заполнения адресных полей всех управляющих МК, обеспечивая тем самым прохождение всех путей на ГСА.

Сравнение рассмотренных трех вариантов реализации УА на основе ПЗУ с принудительной адресацией и двумя адресными полями, с принудительной адресацией и одним адресным полем, с естественной адресацией показывает, что наименьшую разрядность ПЗУ обеспечивает вариант с использованием естественной адресации. При этом время реализации заданного алгоритма оказывается наибольшим, в основном из-за увеличения общего числа выполняемых микрокоманд. Если при выбранном способе адресации объем оборудования построенного УА оказывается чрезмерным или же быстродействие автомата недостаточно, то можно принять некоторые дополнительные меры.

Так, для уменьшения объема ПЗУ находят рациональное разбиение полного множества выходных сигналов yj на подмножества, каждому из которых выделяется свое операционное поле У так, чтобы общее число разрядов операционной части формата МК было наименьшим. Сокращение длины адресной части формата МК можно получить страничной организацией (сегментацией) ПЗУ.

При этом ПЗУ разбивается на сегменты по 2^q ячеек в каждом и адрес каждой формируется из двух частей: из адреса (номера) соответствующего сегмента и адреса ячейки в нем. Специальной микрокомандой адрес сегмента, в пределах которого осуществляется работа, записывается в отдельный регистр или счетчик, а в последующих МК указывается лишь адрес ячейки в сегменте.

Основным средством повышения быстродействия УА является организация опережающей выборки микрокоманд, т. е. организация конвейерного режима работы микропрограммного ДУ. При этом процесс выборки и дешифрации следующей МК совмещается во времени с процессом выполнения предыдущей МК в ОА.

Другие возможности повышения быстродействия УА заключаются в параллельной выборке нескольких МК, которые затем обрабатываются в порядке, диктуемом микропрограммой, а также в организации параллельного анализа нескольких логических условий при осуществлении сложных переходов. Однако, эти меры требуют существенного увеличения объема оборудования.

3.8 Общая структура микропроцессорного вычислительного устройства

При создании современной радиоэлектронной аппаратуры используются три основные подхода к реализации дискретных устройств (ДУ): аппаратный, программный и аппаратно-программный. При аппаратном получают ДУ с традиционной «жесткой» логикой, что обеспечивает наибольшее быстродействие устройств, но требует трудоемкой разработки индивидуальной структуры ДУ.

При программном ДУ реализуется в виде программы для готовой универсальной ЭВМ, в качестве которой можно использовать микроЭВМ, предназначенную для встраивания непосредственно в разрабатываемые блоки.

Аппаратно-программный подход предполагает разработку как программных, так и аппаратных средств. Сюда относится реализация ДУ в виде автомата с микропрограммным управлением и хранимой в ПЗУ программой, а также построение ДУ на основе микропроцессора (МП). Этот вариант открывает широкие возможности для применения современных БИС и позволяет в наибольшей степени согласовать разрабатываемые аппаратно-программные средства с особенностями решаемых задач.

Микропроцессор представляет собой функционально законченное цифровое устройство, выполненное в виде одной или нескольких БИС и предназначенное для выполнения операций по обработке информации и управлению в соответствии с хранимой в памяти программой. Необходимо отметить, что термин «микропроцессор», несмотря на широкое распространение, не имеет строгого определения. Это обусловлено прежде всего наличием большого числа сильно различающихся между собой типов МП, а также их постоянным развитием.

В узком смысле МП совпадает с центральным процессорным элементом (ЦПЭ) вычислительного устройства, выполненным на основе БИС. ЦПЭ обычно используется в качестве основного элемента микропроцессорного вычислительного устройства МПВУ, схема которого представлена на рис. 3.10.

МПВУ минимальной конфигурации содержит ЦПЭ, блоки ПЗУ и ОЗУ, генератор тактовых импульсов ГТИ и блок интерфейса (ИФ), через который осуществляется связь с внешними устройствами (ВУ). Будем считать, что МПВУ, представляющее собой специализированное вычислительное устройство, используется в аппаратуре для выполнения некоторого заданного алгоритма обработки информации (или совокупности алгоритмов).

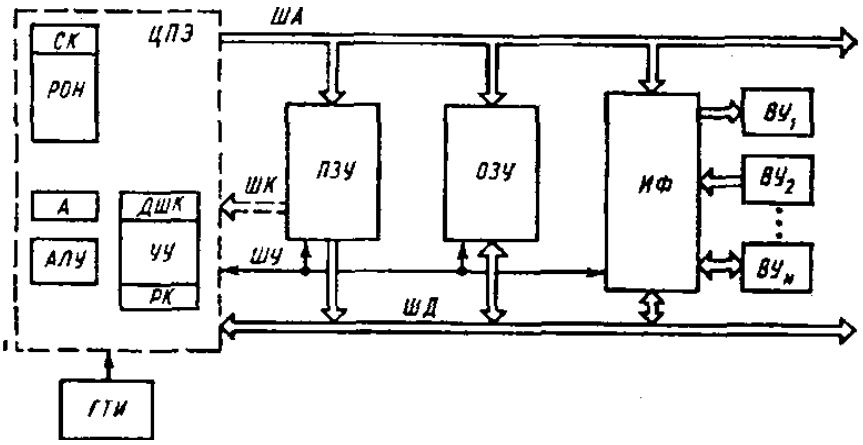


Рисунок 3.10-Обобщенная структурная схема микропроцессорного вычислительного устройства

Поэтому основная программа работы МПВУ записывается в ПЗУ, которое служит также для хранения различных подпрограмм, констант, таблиц и других данных, известных уже на этапе проектирования устройства. ОЗУ используется для хранения данных, поступивших из ВУ или подготовленных для выдачи в ВУ, а также промежуточных результатов вычислений и некоторой адресной информации.

Блок ГТИ, выполняемый, как правило, на основе кварцевого генератора, предназначен для выработки серий тактовых кварцевого генератора, предназначен для выработки серий тактовых импульсов и некоторых вспомогательных сигналов, необходимых для работы ЦПЭ и синхронизации других блоков системы.

Интерфейс представляет собой совокупность шин для передачи информации, электронных схем, специальных сигналов и алгоритмов, управляющих обменом информации. Блок интерфейса служит для сопряжения сигналов МПВУ и ВУ по временным и электрическим параметрам, а также в необходимых случаях для преобразования данных и управления обменом.

К основным узлам ЦПЭ относятся: управляющее устройство (УУ) с регистром команд (РК) и дешифратором команд (ДШК); арифметико-логическое устройство (АЛУ) с аккумулятором (А), который является основным рабочим регистром; блок регистров общего назначения (РОН) со счетчиком команд (СК).

Связь между блоками МПВУ осуществляется с помощью ряда шин: шины адреса (ША), шины данных (ШД), шины управления (ШУ), шины команд (ШК).

Возможны различные варианты организации шин: используется одна двунаправленная шина данных, либо две одно направленные (одна из которых является входной для ЦПЭ, а другая -выходной), шина команд может совмещаться с шиной данных при обеспечении временного разделения сигналов и т. д.

Обобщенно процесс выполнения команды в МПВУ можно разбить на две фазы: фазу выборки кода команды и фазу ее исполнения. Фаза выборки состоит из трех шагов: сначала адрес команды из СК выставляется на ША, затем происходит выборка кода команды из ПЗУ и передача его через ШК или ШД в регистр команд ЦПЭ, после чего производится дешифрация этого кода в ДШК.

В соответствии с кодом команды УУ начинает вырабатывать последовательность управляющих сигналов, необходимых для ее выполнения. Фаза выполнения команды начинается с подготовки операндов (т. е. обрабатываемых данных), которая заключается в определении местоположения операндов и их размещении в требуемых узлах, после чего ЦПЭ переходит к выполнению операции, заданной кодом команды. В это время в СК

формируется адрес следующей команды и вся описанная последовательность работы МПВУ повторяется. Более детально процесс работы МПВУ рассматривается при изучении конкретных серий микропроцессоров.

В зависимости от требований реального применения МПВУ в минимальную конфигурацию системы могут быть введены: контроллер приоритетных прерываний (КПП); контроллер прямого доступа к памяти (КПДП); программируемый параллельный адаптер (интерфейс) ППА); программно-управляемый связной интерфейс (ПСИ); программируемый таймер (ПТ), и т. п.

Блок КПП способствует организации работы МПВУ в реальном времени тем, что дает возможность осуществить временное ВУ, вызвавшего прерывание. Блок КПДП позволяет ускорить обмен массивами данных между ВУ и ЗУ за счет исключения ЦПЭ из цепи передачи информации. Блоки ППИ и ПСИ позволяют организовать обмен между ЦПЭ и В У информацией, представляемой соответственно в параллельном и последовательном кодах. Блок ПТ служит для выработки временных задержек программируемой длительности и меток времени, что способствует организации работы МПВУ в реальном времени.

Для реализации этих блоков во многих микропроцессорных комплектах БИС предусмотрены соответствующие интегральные схемы. Кроме перечисленных типовых блоков в МПВУ могут вводиться нестандартные блоки, специально разработанные для решения конкретных задач.