1 Понятие о триггере

Триггеры и регистры являются простейшими представителями цифровых микросхем, имеющих внутреннюю память. Если выходные сигналы логических элементов и комбинационных микросхем однозначно определяются их текущими входными сигналами, то выходные сигналы микросхем с внутренней памятью зависят также еще и от того, какие входные сигналы и в какой последовательности поступали на них в прошлом, то есть они помнят предысторию поведения схемы. Именно поэтому их применение позволяет строить гораздо более сложные и интеллектуальные цифровые устройства, чем в случае простейших микросхем без памяти. Микросхемы с внутренней памятью называются еще последовательными или последовательностными, в отличие от комбинационных микросхем.

Триггеры и регистры сохраняют свою память только до тех пор, пока на них подается напряжение питания. Иначе говоря, их память относится к типу оперативной (в отличие от постоянной памяти и перепрограммируемой постоянной памяти, которым отключение питания не мешает сохранять информацию). После выключения питания и его последующего включения триггеры и регистры переходят в случайное состояние, то есть их выходные сигналы могут устанавливаться как в уровень логической единицы, так и в уровень логического нуля. Это необходимо учитывать при проектировании схем.

Большим преимуществом триггеров и регистров перед другими типами микросхем с памятью является их максимально высокое быстродействие (то есть минимальные времена задержек срабатывания и максимально высокая допустимая рабочая частота). Именно поэтому триггеры и регистры иногда называют также сверхоперативной памятью. Однако недостаток триггеров и регистров в том, что объем их внутренней памяти очень мал, они могут хранить только отдельные сигналы, биты (триггеры) или отдельные коды, байты, слова (регистры).

Триггер можно рассматривать как одноразрядную, а регистр — как многоразрядную ячейку памяти, которая состоит из несколько триггеров, соединенных параллельно (обычный, параллельный регистр) или последовательно (сдвиговый регистр или, что то же самое, регистр сдвига).

Триггер - одноразрядный элемент памяти, предназначенный для хранения логической переменной (бистабильная ячейка с двумя устойчивыми состояниями). Его схема в отличие от комбинационных схем, рассмотренных в предыдущем разделе, является *последовательностной,* т. е. такой схемой, в которой текущие значения выходов зависят не только от текущих значений входов, до и от работы схемы до текущего момента.

Последовательностные системы выполняют систематические последовательности действий, синхронизуемые управляющим сигналом, называемым *тактовым.* Генератор тактовых импульсов - устройство, генерирующее серию поочередно появляющихся нулей и единиц (последовательность импульсов). Время одного полного цикла тактового сигнала называется *периодом.* Частота тактовых сигналов является

величиной, обратной периоду. Для генерации тактовых сигналов обычно используется нестабильный мультивибратор, работающий на частоте нескольких Мгц.

Триггер является основным компонентом более сложных последовательностных устройств, таких, как счетчики, сдвиговые регистры и регистры памяти, которые в свою очередь являются базовыми для микро-ЭВМ. Хотя известно много разновидностей триггеров, их основу составляет несколько простых схем.

2 RS -триггер

Триггер этого типа имеет два входа, обозначаемых буквами S (от английского set — установить) и R (от reset — сбросить) и используемых соответственно для установки и сброса триггера. Функционирование такого RS-триггера описано в таблице 1. Обратите внимание, что триггер оказывается в состоянии логической 1 (установлен), когда логическая 1 подается только на вход S, и в состоянии логического 0 (сброшен), когда логическая 1 подается только на вход R. Состояние триггера не меняется, если на оба входа подаются логические 0. По этой причине о входной комбинации $S=0$ и $R=0$ говорят как о случае отсутствия входных сигналов. И наконец, поведение триггера для комбинации $S=1$ и $R=1$ не определено.

Простой RS-триггер можно сконструировать, соединив «крест-накрест» два вентиля И-НЕ, как показано на рис. 2.20. Вентили могут быть выполнены по любой технологии: ТТЛ, КМОП и т.п. Состояние триггера соответствует значению выходной переменной Q на верхнем вентиле И-НЕ. В дальнейшем мы увидим, что на выходе нижнего вентиля И-НЕ будет значение Q, являющееся дополнением Q. Входные линии вентиля обозначены S и R, поскольку триггер устанавливается при $S=0$ и сбрасывается при $R=0$.

Чтобы удостовериться, что наша схема действительно ведет себя в соответствии с определением RS-триггера, рассмотрим сначала комбинацию входов $\overline{S}=1$ и $\overline{R}=1$, соответствующую отсутствию входных сигналов. В этом случае перекрестные связи между двумя вентилями И-НЕ приведут к поддержанию одного из двух возможных состоянии.

Функционирование RS-триггера

Q обозначает текущее состояние триггера, а Q*—результирующее состояние, получающееся при заданной комбинация входов.

Таблица 1

S	R	Q*
0	0	Q
0	1	0
1	0	1
1	1	Не определено

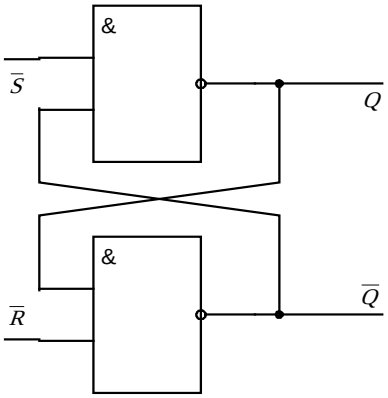


Рис. 1. Простой RS - триггер

В частности, если на выходе линии Q будет логическая 1, то на обоих входах нижнего вентиля будут логические 1, приводящие к логическому 0 на выходе $\overline{Q}$. Этот логический 0, поданный на один из входов верхнего вентиля, вызовет логическую 1 на его выходе Q, что и было предположено в начале нашего рассуждения.

С другой стороны, если на выходе Q логический 0, то он подается на вход нижнего вентиля, в результате чего выход $\overline{Q}$ оказывается равным 1. На обоих входах верхнего вентиля при этом окажутся единицы, что приведет к нулевому значению на выходе Q, а это также соответствует начальному предположению. Таким образом, мы можем заключить, что любое из двух возможных состояний триггера стабильно, когда на входы подается комбинация $\overline{S}=0$ и $\overline{R}=0$.

Теперь рассмотрим две другие входные комбинации, для которых поведение триггера определено. Для каждой из этих комбинаций стабильным оказывается только одно из возможных состояний триггера. В частности, пусть входная комбинация есть $\overline{S}=0$ и $\overline{R}=1$. Выходное значение Q верхнего вентиля при логическом 0 на входе S будет равно 1. Теперь логические 1 на обоих входах нижнего вентиля дадут логический 0 на его выходе $\overline{Q}$. Этот логический 0, поданный на вход верхнего вентиля, будет поддерживать значение логической 1 на его выходе Q даже после того, как S снова примет значение 1.

Таким образом, входная комбинация $\overline{S}=0$ и $\overline{R}=1$ вызовет установку триггера в соответствии с определением. В силу симметрии сразу видно, что входная комбинация $\overline{S}=1$ и $\overline{R}=0$ приведет к логической 1 на выходе $\overline{Q}$ и, следовательно, к логическому 0 на выходе Q. Как и раньше, выходы не изменяются после того, как R вернется к 1. Таким образом, комбинация $\overline{S}=1$ и $\overline{R}=0$ вызывает сброс триггера, что также соответствует определению.

Наконец, заметим, что оставшаяся входная комбинация $\overline{S}=0$ и $\overline{R}=0$ даст логические 1 на выходах обоих вентилей. Это приведет к непредсказуемым результатам после возврата к состоянию отсутствия

входных сигналов, поскольку результирующее состояние окажется зависящим от тепловых шумов, разбросов параметров электронных компонентов и т. п. Поэтому комбинация $\overline{S}=0$ и $\overline{R}=0$ считается недопустимой. Следует обратить внимание на то обстоятельство, что во всех рассмотренных нами случаях, кроме случая недопустимой входной комбинации, выход $\overline{Q}$ оказывался инверсией, или дополнением, выхода Q.

На рис. 2. показано графическое обозначение такого триггера, часто используемое на логических диаграммах. На входных и выходных линиях, так же как и для вентилей, могут быть кружочки, обозначающие инверсию сигнала. В частности, кружочек на синхронизирующем входе говорит о том, что триггер открыт при нулевом значении синхросигнала, а не при единичном.

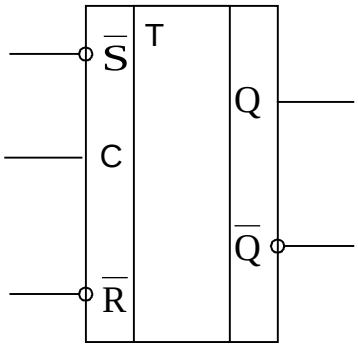


Рис. 2. Синхронный RS – триггер.

3 Пример RS-триггера

Микросхема ТР2 является триггером RS с парафазными входами для записи информации. Триггер ТР2 выполнен на двух логических элементах 2И—НЕ с обратными связями. Назначение входов R и S выбрано относительно выхода Q так, что записываемая информация соответствует 1 высокому или низкому уровню напряжения с учетом инверсии входных сигналов R и S. Микросхема ТР2 включает четыре триггера. Для расширения функциональных возможностей ИС два триггера из четырех имеют по два входа S, логически объединенных по И, в отрицательной логике по ИЛИ. Достаточно на одном из входов S триггера установить низкий уровень напряжения, а на входе R разрешить запись 1 соответствующей информации высоким уровнем и триггер установится в состояние высокого уровня. Установка триггера в состояние высокого или низкого уровня осуществляется кодом 01 или 10 на входах S и R со сменой кода информации триггер является асинхронным. Временная диаграмма его работы приведена на рис. 3

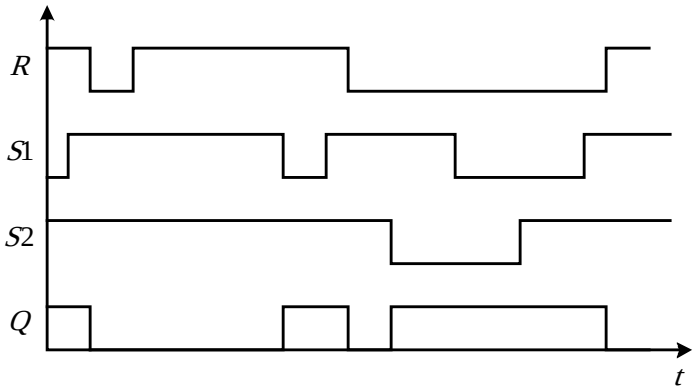


Рис 3. Временная диаграмма работы микросхемы ТР2

3 Триггеры типов JK, D и T

Наряду с *RS* – триггерами широкое распространение получили триггеры, обладающие другими характеристиками. Их можно получить из двухтактного *RS* триггера, добавляя в схему некоторое количество вентилей. На рис. 4. показано, те модификации, какие нужно сделать для получения триггеров трех распространенных типов: JK, D и T.

JK – триггер представляет собой обобщенную версию RS – триггера. Вход J соответствует входу S (т.е. по нему триггер устанавливается), а вход K соответствует входу R (т.е. по нему триггер сбрасывается). Разница в поведении RS и JK триггеров обнаруживается, когда на оба входа подается логическая 1. В то время как поведение для входной комбинации $S=1$, $R=1$ RS – триггера не определено, поведение JK – триггера определено. А именно если на вход подана комбинация $J=1$, $K=1$, JK – триггер по синхроимпульсу изменяет свое состояние на противоположное.

Схема двухтактного JK – триггера показана на рис. 4(а). Она отличается от схемы двухтактного RS – триггера двумя вентилями И на входах R и S. Эти вентили блокируют воздействие одного из входов J или K в зависимости от состояния триггера. Блокируется та линия, единичное значение на которой не вызывало бы изменения состояния триггера. В частности, J блокируется в состоянии, когда на выходе Q логическая 1, а, K блокируется, когда на выходе Q логический 0. Действительно, значение J логически умножается на $\overline{Q}$, а, K — на Q. Следовательно, когда на обоих входах J и K логическая 1, срабатывает только тот вход, который вызывает изменение состояний, и по синхроимпульсу триггер действительно изменит состояние.

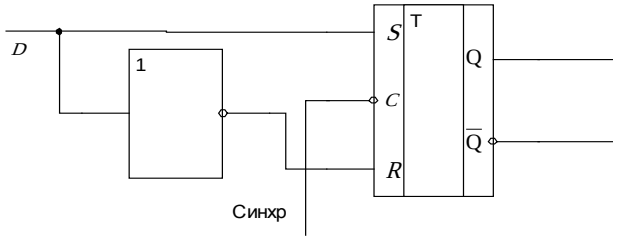
Для триггера типа D соотношение между сигналом на входной линии и следующим состоянием триггера формулируется проще, чем для любого другого типа триггера. По синхроимпульсу D-триггер принимает то состояние, которое имеет входная линия D.

Как показано на рис. 4 (б), D-триггер можно реализовать, соединив S – вход двухтактного RS – триггера непосредственно с входной линией D, а вход R — с инверсией (дополнением) D. При таком соединении, если на D логическая 1, то и на S будет логическая 1, и триггер по синхроимпульсу

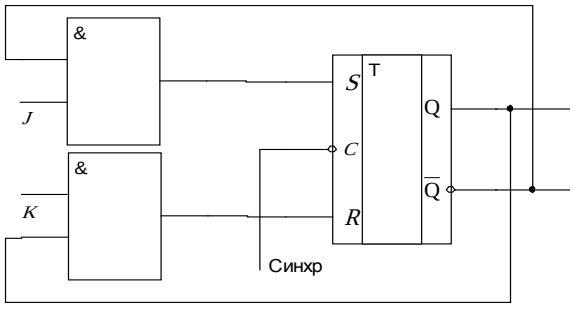
будет установлен. Если же на D логический 0, то на R будет 1, и триггер по синхроимпульсу будет сброшен.

Соотношение между состоянием Т-триггера и его входом довольно интересно и, как мы увидим в дальнейшем, полезно в некоторых приложениях, особенно при реализации счетчиков. Если входная линия Т имеет значение 1, то по синхроимпульсу триггер изменит свое состояние на противоположное. В противном случае его состояние сохранится неизменным.

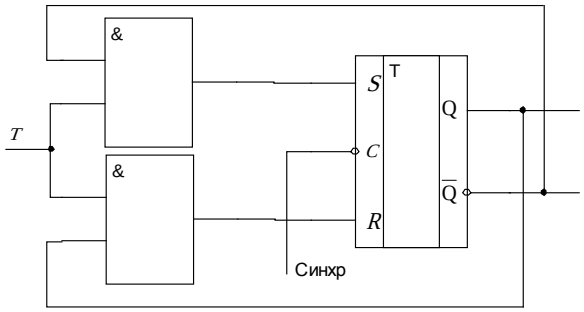
На рис. 4(в) представлена одна из возможных реализации Т – триггера на базе двухтактного RS – триггера. Она отличается от реализации JK – триггера на рис. 4(а) только объединением входов J и K, в один вход Т.



(б)



(а)



(в)

Рис. 4. Триггеры распространенных типов, получаемые из двухтактного RS - триггера (а) JK – триггер, (б) D – триггер (в) Т – триггер.

Функционирование триггеров всех трех описанных типов сведено в табл.2.

Во всех случаях Q обозначает состояние триггера до прихода синхросигнала, а Q* - после, (а) JK-триггер, (б) D - триггер. (в) Т - триггер.

Таблица 2.

J	K	Q*
0	0	Q

D	Q*
0	0

T	Q*
0	*Q*

0	1	0
1	0	0
1	1	$\overline{Q}$

(а)

1	1

(б)

1	$\overline{Q}$

(в)

4 Примеры JK-триггеров

Триггер ТВ1 типа JK (рис. 5.) состоит из двух триггеров: основного (элементы Е7, Е8) и вспомогательного (элементы Е3 – Е6). С приходом фронта импульса синхронизации на вход С происходит запись информации во вспомогательный триггер в соответствии с логическими уровнями сигналов на входах J и К. Одновременно импульс синхронизации блокирует цепи перезаписи информации из вспомогательного триггера в основной. На время действия импульса синхронизации в основном триггере сохраняется информация, записанная в предыдущем такте. С приходом среза импульса синхронизации блокировка снимается и происходит перезапись информации из вспомогательного триггера в основной. Запись информации в JK-триггер происходит по фронту, а изменение состояния на выходе - по срезу импульса синхронизации. Временная диаграмма его работы приведена на рис. 6.

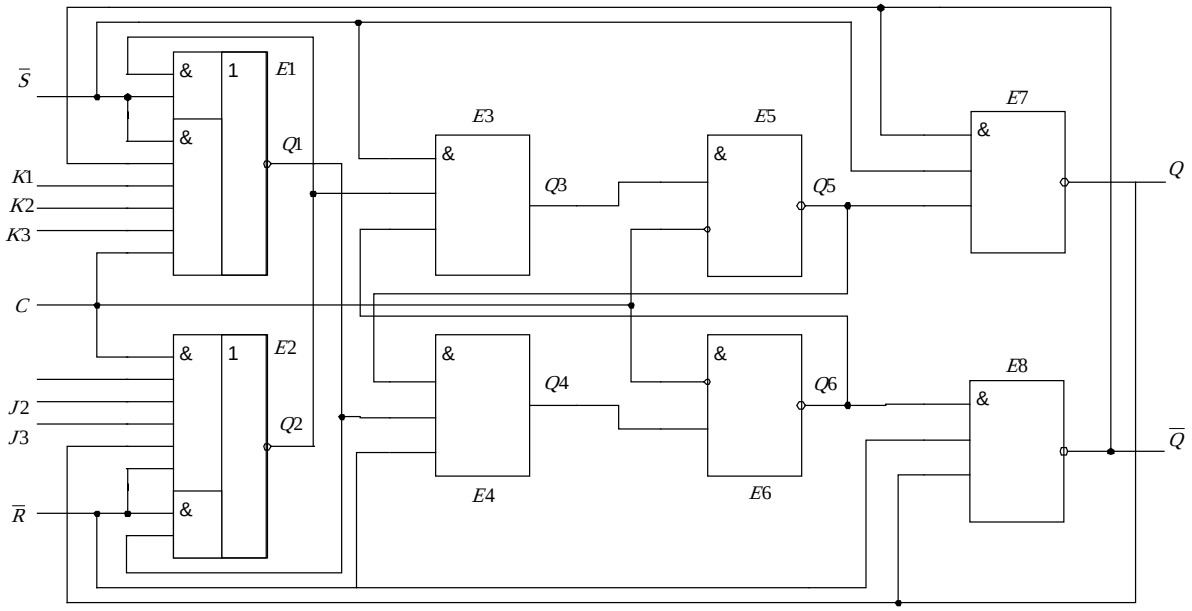


Рис. 5. Логическая структура микросхемы ТВ1.

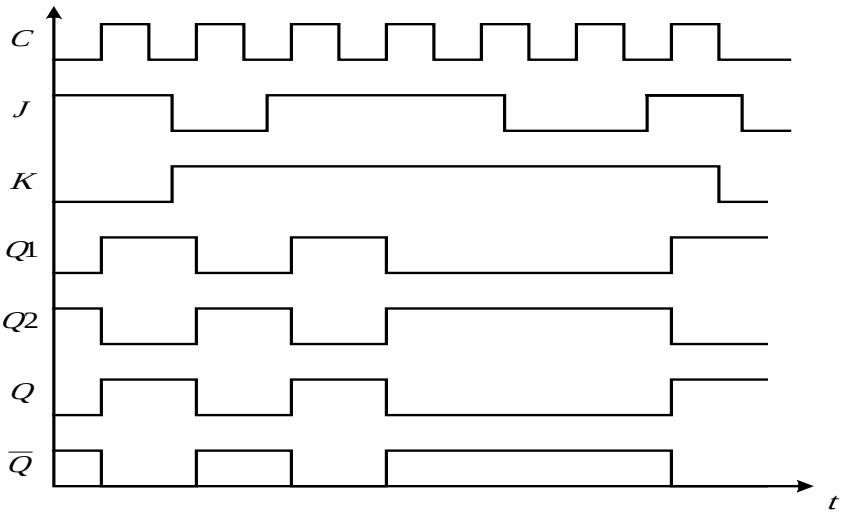


Рис. 6. Временная диаграмма работы микросхемы ТВ1.

В асинхронном режиме при управлении по входам R и S JK-триггер работает аналогично RS-триггеру (состояния входов J, K, C - произвольны). Режим работы JK-триггера в синхронном режиме иллюстрирует табл. 3.

Таблица 3.

$t_п$				t+1	
Выходы		Входы		Выходы	
Q	$\overline{Q}$	J	K	Q	$\overline{Q}$
0	1	0	0	0	1
0	1	0	1	0	1
0	1	1	0	1	0
0	1	1	1	1	0
1	0	0	0	1	0
1	0	0	1	0	1
1	0	1	0	1	0
1	0	1	1	0	1

Пример D–триггера

Микросхема ТМ2 является универсальным D – триггером с однофазным приемом информации и с независимой установкой в состояние низкого и высокого уровней. Логическая структура D – триггера (рис. 7) содержит следующие элементы: основной асинхронный RS – триггер (Т3); вспомогательный синхронный RS – триггер записи высокого уровня в основной триггер (Т1); вспомогательный синхронный RS – триггер записи низкого уровня в основной триггер (Т2).

Запись информации в D – триггер происходит по фронту импульса синхронизации. С приходом фронта импульса синхронизации в момент времени t информация, поступающая на вход D, принимается во вспомогательные триггеры Т1,Т2, но на входе появляется с задержкой в

момент времени t+1: $Q(t+1)=D(t)$. Таким образом D – триггер следит за изменением информации в момент прихода фронта импульса синхронизации.

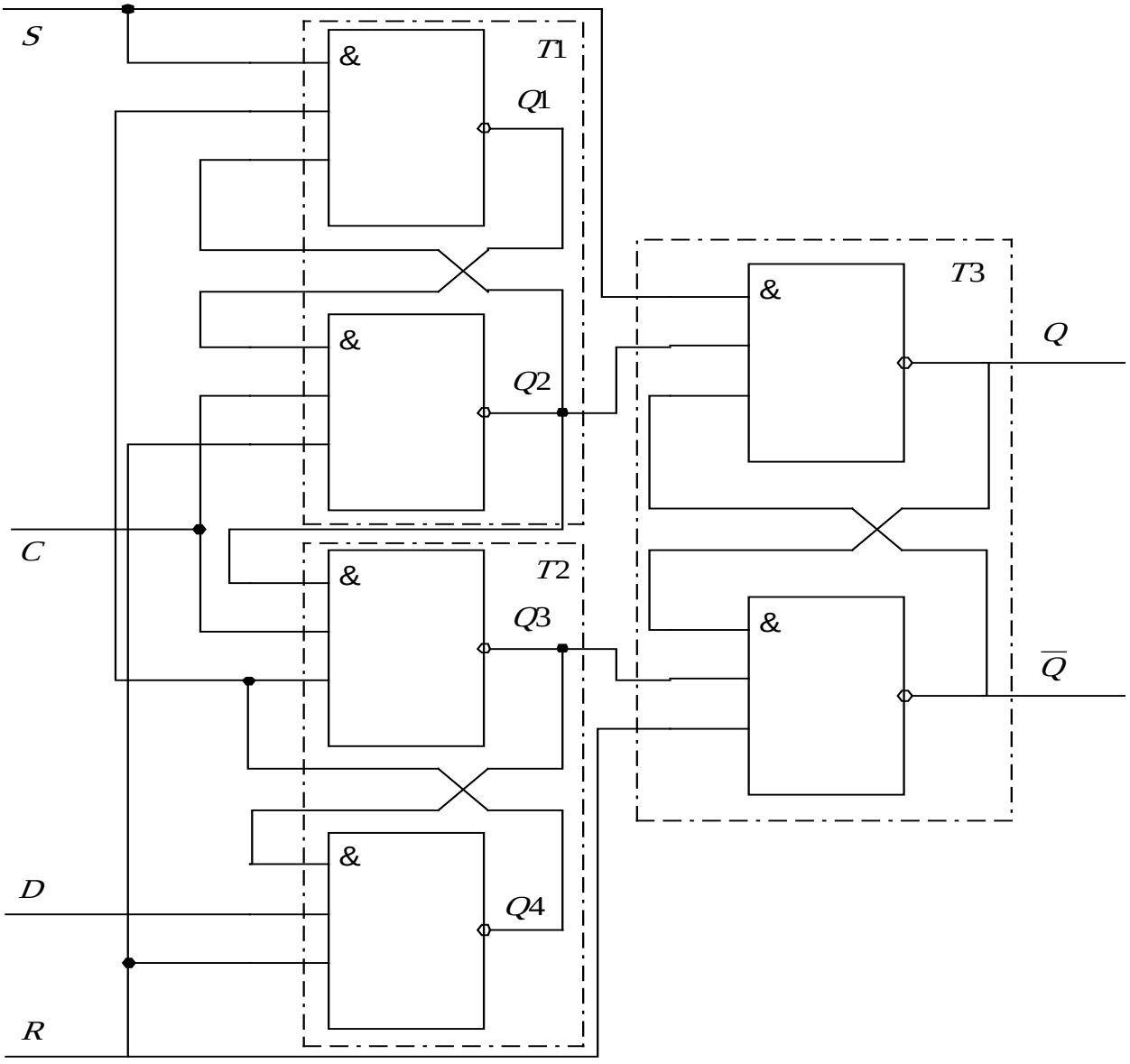


Рис. 7. Логическая структура микросхемы ТМ2

Литература: 2, стр. 278-286; 4, стр.45-56

Контрольные вопросы:

1 Дайте определение триггеру.
2 Что такое RS –триггер?
3 Что такое триггеры типов JK, D ?
4 Примеры JK-триггеров

Тема 7 Графическое изображение триггеров, таблица переключений и построение их на логических элементах.

Цель: Рассмотреть графическое изображение триггеров, таблиц переключений и построение их на логических элементах. Освоить анализ работы по временным диаграммам

План:

1 Триггеры, переключаемые фронтом синхроимпульса
2 Сдвиговые регистры
3 Сдвиговый регистр с последовательным входом.
4 Сдвиговый регистр с параллельным входом
5 Пример регистра

1 Триггеры, переключаемые фронтом синхроимпульса

Одно из свойств двухтактного D-триггера, показанного, состоит в том, что как опрос входных сигналов, так и обновление выходных происходит по фронту (скажем, отрицательному) синхроимпульса. Для этого типа триггера состояние ведущей секции реагирует на изменения входного сигнала все время, пока синхроимпульс имеет значение 1. То значение, которое будет иметь вход D непосредственно перед отрицательным фронтом синхроимпульса, определит окончательное состояние ведущей секции.

Именно это состояние будет передано ведомой секции сразу после отрицательного фронта синхроимпульса. Следовательно, входной сигнал должен иметь правильное значение только в течение небольшого отрезка времени (равного времени срабатывания ведущей секции) непосредственно перед отрицательным фронтом синхроимпульса. О триггерах, обладающих таким свойством, говорят как о триггерах, переключаемых фронтом синхроимпульса.

Ранее описанные триггеры RS, JK и T не обладают этим свойством, поскольку окончательное состояние ведущей секции зависит от значения входного сигнала на всем интервале времени, когда синхроимпульс равен логической 1. Для RS – триггера окончательное состояние ведущей секции зависит от того, какой из входов R или S, последним принимал значение 1 за время действия синхроимпульса.

В случае JK – триггера окончательное состояние ведущей секции зависит от того входа, J или K, который не заблокирован состоянием ведомой секции. Например, когда триггер в состоянии 1, только вход K может вызвать изменение состояния. Если K примет значение 1 в любой момент, когда на синхровходе логическая 1, то состояние ведущей секции станет равной 0. В течение всего интервала вход J заблокирован состоянием ведомой секции (которое все еще равно 1).

Таким образом, состояние ведущей секции, став однажды равным 0, уже не может измениться до следующего синхроимпульса. Это свойство JK – триггера называют захватом нуля. Аналогично JK – триггер, находясь в

нулевом состоянии, демонстрирует свойство захвата единицы. Т – триггер обладает аналогичным свойством захвата нуля и единицы, поскольку он реализуется фактически той же самой схемой, что и JK – триггер.

Поскольку двухтактные RS -, JK – и Т – триггеры не являются переключаемыми фронтом, о значениях входов нужно заботится на протяжении всего интервала, когда на синхровходе 1. В тех случаях, когда требования к входным сигналам слишком обременительны, используются разновидности RS -, JK -, и Т – триггеров, переключаемых фронтом.

Схемы этих разновидностей можно получить из двухтактного D – триггера, подключая дополнительные вентили. Для этого функции входа D определяются через входные сигналы и состояния выходов триггера. Например, JK – триггер можно получить, используя функцию заданную в таблице 1. В этой таблице значение D равно требуемому следующему состоянию триггера для каждой комбинации J, K и текущего состояния Q. Описываемую таблицей функцию можно записать в виде $D = J\overline{Q} + \overline{K}Q$. Схема JK – триггера, переключаемого фронтом, полученная по этой функции, показана на рис. 1. Поскольку значения J и K непосредственно перед отрицательным фронтом синхроимпульса определяют значение D и поскольку D – триггер переключается фронтом, то и весь JK триггер окажется переключаемым фронтом.

Логическая функция для превращения D – триггера, переключаемого по фронту, в JK – триггер, также переключаемого по фронту.

Таблица 1.

J	K	Q	D
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

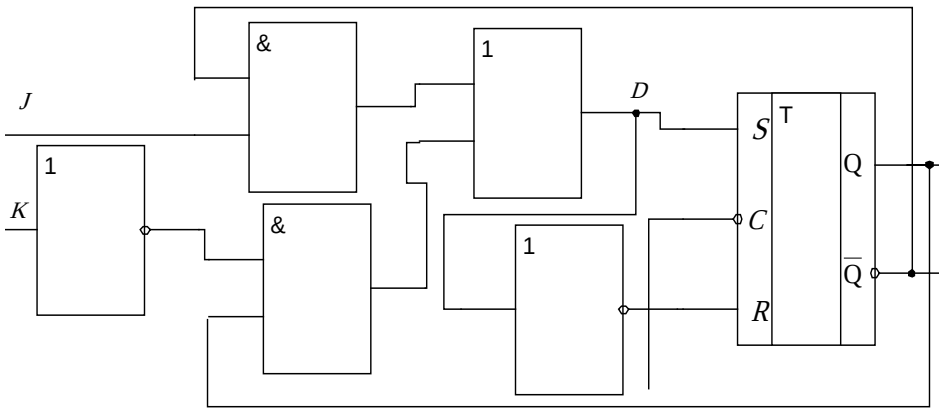


Рис 1. Переключаемый фронтом JK – триггер.

2 Сдвиговые регистры

Для хранения и обработки информации в микро-ЭВМ широко используются сдвиговые регистры. Сдвиговый регистр состоит из ряда триггеров (по одному на каждый бит информации), соединенных так, что выход каждого триггера подключен ко входу следующего. Информация в регистре сдвигается на один разряд вправо или влево при поступлении каждого тактового импульса. Это устройство идеально подходит для обработки последовательной информации (подаваемой по биту в каждый момент времени), преобразования параллельной информации (все биты поступают одновременно) в последовательную и последовательной в параллельную.

Сдвиговые регистры реализуются на СИС – устройствах, выполненных с применением RS-, JK-, или D – триггеров, и различия между ними главным образом связаны с методом обработки входных и выходных данных. В данном разделе описываются основные типы этих регистров.

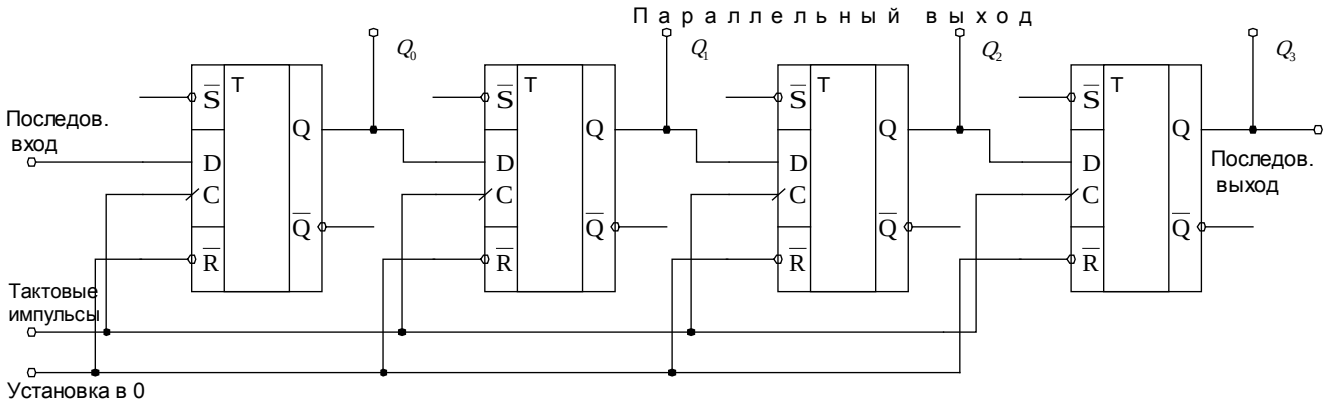


Рис. 2. Типичный 4^x разрядный регистр с последовательным входом.

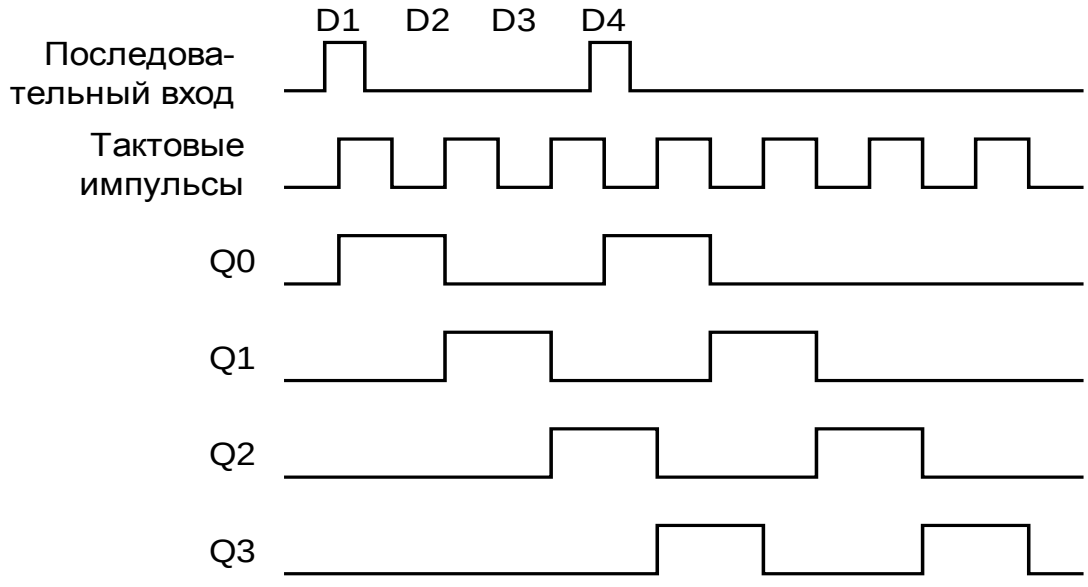


Рис. 3. Временная диаграмма работы 4^x разрядного сдвигового регистра.

3 Сдвиговый регистр с последовательным входом.

Сдвиговый регистр с последовательным входом – это устройство, в котором данные последовательно поступают на вход, как показано на рис. 2 для 4^x разрядного сдвигового регистра. В данном случае используются D – триггеры. Работает регистр следующим образом. В исходном положении импульс сброса (логический 0) подается на вход «Установка в 0», устанавливая выходы Q_0-Q_3 в 0. Дале первый бит данных подается на последовательный вход. При воздействии переднего фронта первого тактового импульса Q_0 принимает значение равное D_1. Затем на последовательный вход подается D_2. При воздействии переднего фронта второго тактового импульса $Q_0=D_2$ и $Q_1=D_1$. Продолжается этот процесс, после четырех тактовых импульсов имеем $Q_0=D_4$, $Q_1=D_3$, $Q_2=D_3$, $Q_3=D_1$. Временная диаграмма для последовательно поступающих входных данных показана на рис. 3.

Выход данных при этом может быть как последовательным так и параллельным. В последнем случае сдвиговый регистр работает как последовательно-параллельный преобразователь. Очевидно, для сдвиговых регистров, имеющих большое число разрядов (более восьми), параллельные выходы нецелесообразны из-за большого количества выходов в корпусе ИС. Существуют сдвиговые регистры, имеющие более 1000 разрядов.

4 Сдвиговый регистр с параллельным входом

Сдвиговый регистр с параллельным, входом — это устройство, в котором входные данные поступают одновременно по параллельным информационным каналам (рис. 4). Запись данных в регистр осуществляется следующим образом. Сначала производится сброс содержимого регистра подачей импульса (логического 0) на вход «Установка в 0». Далее D_1—D_4 подаются на входы и импульс (логическая 1) .поступает на вход записи. Это приводит к записи информации во все регистры с использованием входов предустановки. После этого при появлении каждого тактового импульса информация сдвигается на один разряд вправо. Выход данных может быть как последовательным, так и параллельным. Многие сдвиговые регистры, выполненные в виде ИС, имеют параллельный вход и последовательный выход. Эти устройства известны как параллельно-последовательные преобразователи.

В описанных выше сдвиговых регистрах сдвиг производился в одном направлении при появлении каждого тактового импульса. Во многих случаях, однако, желательно иметь возможность сдвигать информацию и влево, и вправо. Регистры, обладающие этой способностью, называются реверсивными сдвиговыми регистрами. Управление сдвигом в таких регистрах осуществляется путем подключения выходов триггеров к

соответствующим входам при сдвиге влево или вправо. Направление сдвига регулируется входом «Способ работы». Реверсивные сдвиговые регистры с последовательными и параллельными входами и выходами называют универсальными сдвиговыми регистрами.

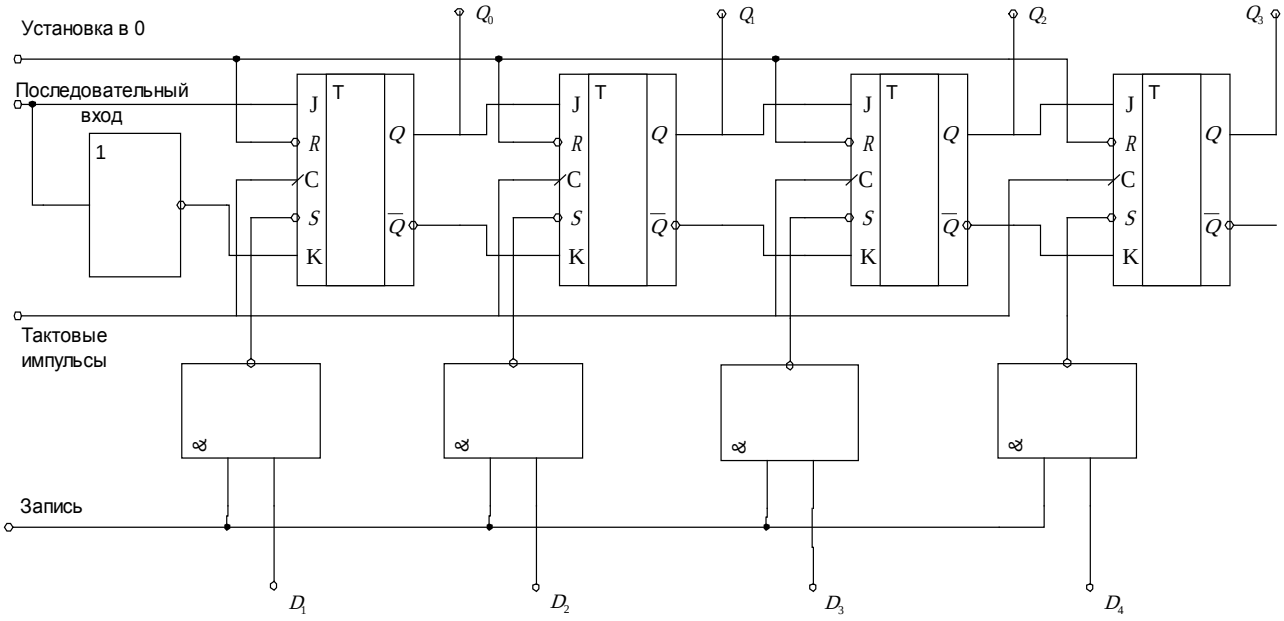


Рис. 4. Типичный 4-разрядный сдвиговый регистр с параллельным выходом.

5 Пример регистра

В микросхеме ИР1 каждый разряд образован синхронным двухступенчатым триггером RS с логикой на входе (рис. 5). Регистр сдвига позволяет реализовать следующие режимы работы: запись информации параллельным кодом; сдвиг вправо; сдвиг влево. Управление режимом работы регистра осуществляется по входам VI, V2, C1, C2 (выводы 1, 6, 9, 8).

Для записи в регистр информации параллельным кодом следует на вход управления режимом V2 подать напряжение высокого уровня, на вход C2 напряжение низкого уровня, а информационные сигналы на входы D1 — D8. Напряжение на входах C1, VI может быть любым. Для сдвига за писанной параллельным кодом информации вправо тактовые импульсы подаются на вход C2 (вывод 8). При этом на входе V2 (вывод 6) следует поддерживать напряжение высокого уровня. При операциях с данными, представленными в последовательном коде, входную информацию в виде последовательности импульсов подают на вход информации VI (вывод 1), тактовые импульсы на вход синхронизации C1 (вывод 9), а на входах V2, D1 — D8 поддерживают напряжение низкого уровня. Режимы работы ИС ИР1 при различных видах записи информации представлены в табл. 1.

Таблица 1

Режим работы	Сигналы управления		
	V2	C1	C2

Запись параллельным кодом	1	X	1/0
Запись последовательным кодом (сдвиг)	0	1/0	X

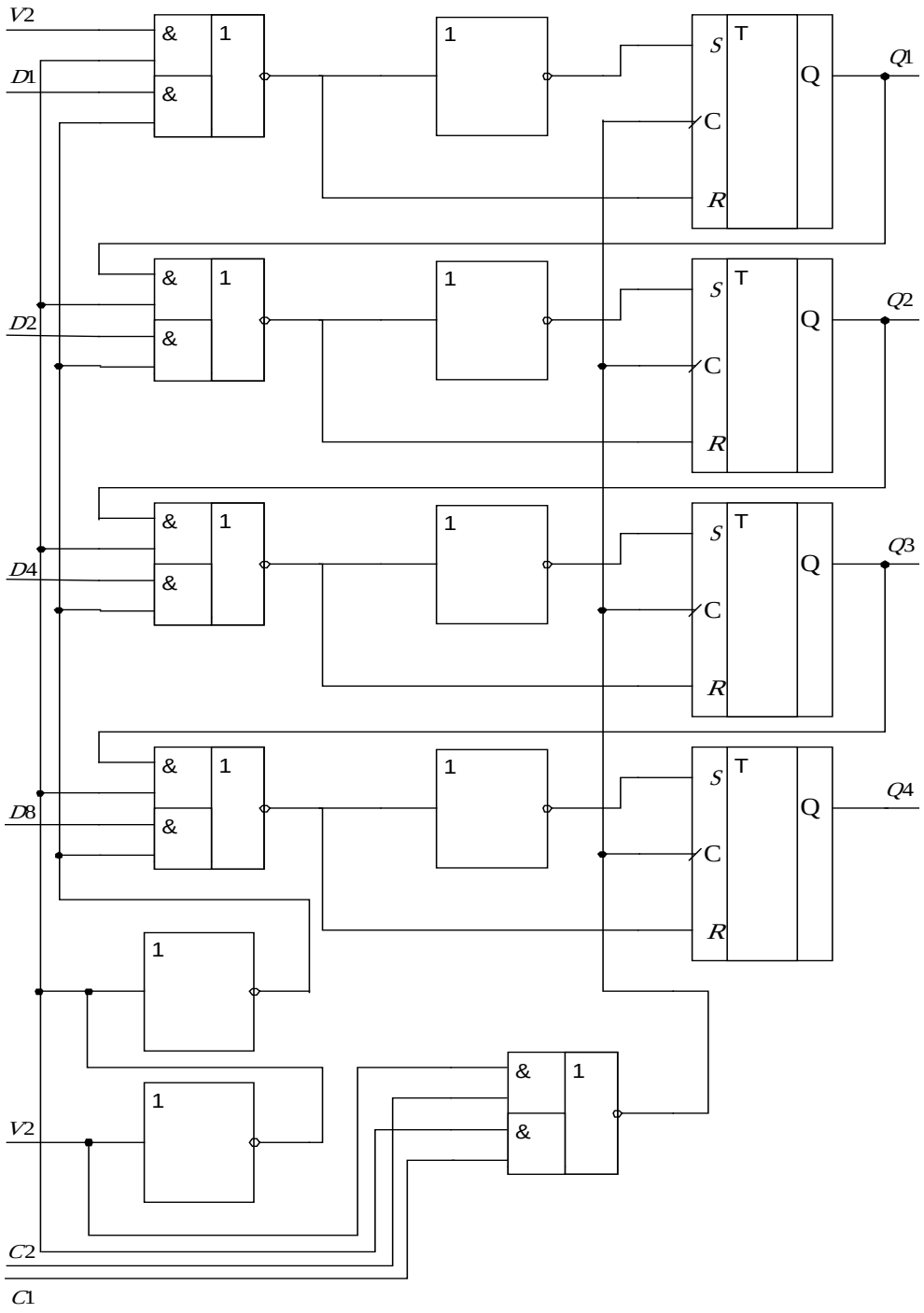


Рис. 5. Логическая структура микросхемы ИР1

При сдвиге влево на вход выбора режима V2 подается напряжение высокого уровня, которое блокирует прохождение тактовых импульсов, для сдвига вправо. Если при этом на входы параллельного кода разрядов D1 — D8 не подавать параллельный код числа, а выход последнего разряда соединить с входом параллельного кода предыдущего разряда, его выход с

аналогичным входом предшествующего ему разряда и т. д. то получим регистр сдвига влево. Входом последовательного кода в этом случае служит вход параллельного кода последнего разряда регистра сдвига.

Микросхемы ИР1 могут быть использованы в качестве основного элемента в арифметических устройствах буферной памяти, элемента задержки на n тактов, преобразователя последовательных кодов в параллельные и наоборот, делителя частоты, закольцованного распределителя импульсов и т. д.

Литература: 1, стр.148-156; 2, стр. 189-194; 3, стр. 85-89

Контрольные вопросы

1 Что представляют собой триггеры с синхроимпульсом?
2 Что такое сдвиговый регистр?
3 Как выглядит 4-разрядный сдвиговый регистр с параллельным выходом?
4 Как выглядит логическая структура микросхемы ИР1?

Тема 8 Основные понятия теории автоматов. Типы конечных автоматов: автоматы Мура и Мили.

Цель: Ввести понятие теории автоматов. Рассмотреть автоматы Мура и Мили.

План:

1 Основные понятия теории автоматов
2 Типы конечных автоматов: автоматы Мура и Мили.
3 Микропрограммные автоматы на ПЗУ

1 Основные понятия теории автоматов

В настоящее время при программной реализации алгоритмов логического управления технологическими процессами наряду с программируемыми логическими контроллерами все чаще используются компьютеры, предназначенные для работы в производственных условиях.

Однако в последнее время специалисты по проектированию программ обратили внимание на то, что только структурное проектирование указанной проблемы не решает и предложили объектно-ориентированный подход к проектированию программ, в рамках которого было введено понятия "объекта" и его "состояния" и рекомендовано использовать графы переходов для описания динамики реализуемых процессов.

В данной теме рассматриваются требования к построению легко понимаемых граф-схем и графов переходов, и предлагаются методы построения понимаемых графов переходов по граф-схемам алгоритмов и понимаемых граф-схем по графам переходов, а также методы их программирования в базисе языков различных уровней.

Граф-схемы. Основные проблемы.

Будем рассматривать в настоящей работе автоматные граф-схемы, в которых в операторных вершинах формируются или сохраняются только единичные и нулевые значения переменных. Автоматные граф-схемы разобьем на два класса: граф-схемы алгоритмов и граф-схемы программ.

При этом будем различать граф-схемы алгоритмов (ГСА) по следующим основным признакам:

- наличию внутренних обратных связей;
- используемым типам переменных;
- наличию дешифратора состояний;
- наличию умолчаний и неоднозначно задаваемых переменных;
- наличию переменных, которые могут неоднократно изменяться за один проход граф-схемы;
- месту выдачи значений выходных переменных.

Используя некоторые из этих признаков, выделим пять подклассов граф-схем алгоритмов:

1 граф-схемы алгоритмов с внутренними обратными связями и выдачей значений выходных переменных в любой операторной вершине, обозначаемые ГСА1;

2 граф-схемы алгоритмов без внутренних обратных связей и дешифратора состояний, в которых выдача значений выходных переменных выполняется в конце "тела" граф-схемы, обозначаемые ГСА2;

3 граф-схемы алгоритмов без внутренних обратных связей, учитывающие особенности используемых управляющих конструкций языка программирования, обозначаемые ГСА3;

4 граф-схемы алгоритмов без внутренних обратных связей, но с дешифратором состояний и выдачей значений выходных переменных в конце "тела" граф-схемы, которые не содержат выходных переменных, неоднократно изменяющихся за один проход граф-схемы, обозначаемые ГСА4;

5 граф-схемы алгоритмов без внутренних обратных связей, но с дешифратором состояний и выдачей значений выходных переменных в конце "тела" граф-схемы, содержащие выходные переменные, которые могут неоднократно изменяться за один проход граф-схемы, обозначаемые ГСА5.

При этом отметим, что внешняя обратная связь в управляющих алгоритмах и программах существует всегда (режим сканирования в программируемых логических контроллерах). Отметим также, что если в вычислительном устройстве реализуется алгоритм управления в виде одной компоненты (задачи), то ГСА1, используя ГСА3, могут при наличии соответствующих управляющих конструкций изоморфно отражаться в граф-схемы программ.

На рис.1 в качестве примера приведена схема связи управляющего автомата с объектом управления, состоящим из трех клапанов (Кл1, Кл2,

Кл3) и двигателя (Д). При этом управляющий автомат декомпозирован на автомат (А) и функциональные элементы задержки (ФЭЗ).

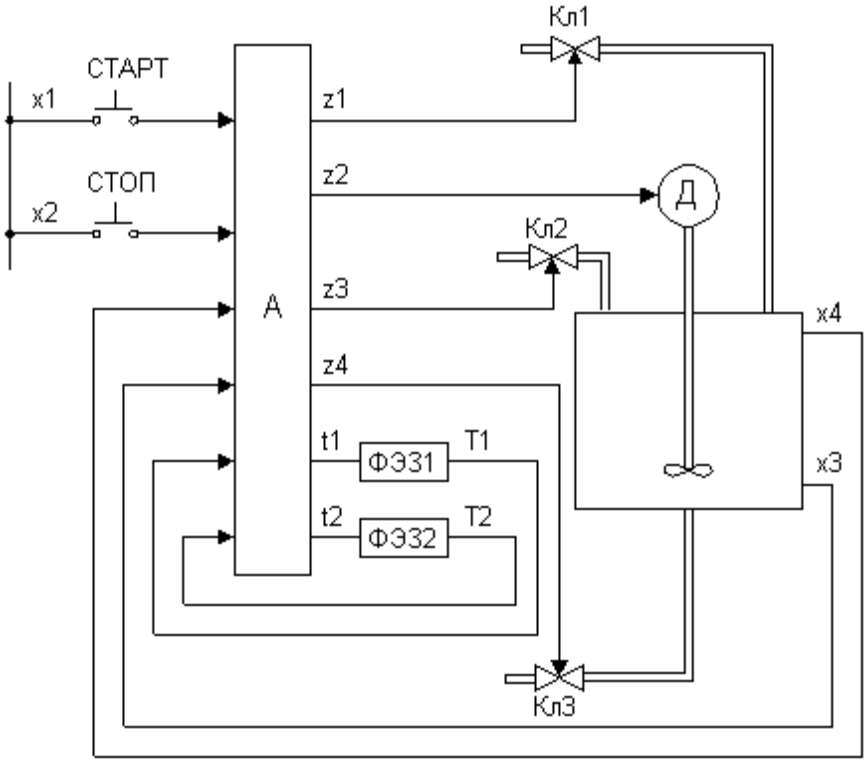


Рис. 1. Схема связи управляющего автомата с объектом управления из трех клапанов.

2 Типы конечных автоматов: автоматы Мура и Мили. Графы переходов. Классификация

Определение 1. Граф переходов, в котором в каждой вершине явно указаны значения каждой выходной переменной, назовем графом переходов автомата **Мура** с явным заданием значений всех выходных переменных.

В этом случае значения выходов соответствуют номеру вершины и не зависят от предыстории. По этой причине такой граф переходов просто понимается и в него легко вносятся изменения.

Определение 2. Граф переходов, в котором в вершинах, кроме явно определенных значений одних переменных, применяются также неявно, но однозначно определенные значения других переменных, назовем графом переходов автомата **Мура** с неявным заданием значений выходных переменных.

Неявное задание переменной в вершине отображается прочерком, который в данном случае может быть заменен только одним значением булевой переменной. Эта переменная в рассматриваемой вершине сохраняет то значение, которое присваивается ей во всех "смежных" с ней вершинах. Связь между двумя вершинами может быть непосредственной с помощью заходящей в вершину дуги или транзитной через другие вершины, в которых вместо значений этой переменной используются проверки.

Графы переходов этого типа читаются несколько хуже, чем графы переходов автоматов **Мура** первого типа, однако их целесообразно

использовать для сокращения объема памяти программ, изоморфно отражающих графы переходов автоматов **Мура**. Графы переходов этого класса могут быть изображены таким образом, что в нем в каждой вершине в явном виде указаны значения каждой выходной переменной, а неизменяющиеся относительно "смежных" вершин значения этих переменных отмечаются перечеркиванием.

Определение 3. Граф переходов, в котором в вершинах, кроме явно определенных значений одних переменных, используются также неявно и неоднозначно заданные значения других переменных, назовем графом переходов автомата **Мура** с неоднозначным заданием значений выходных переменных.

Графы переходов этого класса могут содержать для одной и той же задачи меньшее число вершин, чем графы переходов автоматов Мура двух других указанных выше разновидностей. В этом случае в вершине вместо одного значения умалчиваемой переменной могут формироваться различные значения той же переменной, что порождает в этой вершине различные наборы значений выходных переменных и обеспечивает эквивалентность одной такой вершины нескольким, полностью определенным.

Это преимущество с точки зрения компактности описания и сокращения длины программы порождает одновременно и главный недостаток графов переходов этого типа — трудность их чтения и понимания. Именно по этой причине такие графы переходов не рекомендуется использовать в качестве языка общения.

Спецификацию задач рассматриваемого класса целесообразно производить с помощью двух первых моделей графов переходов автоматов **Мура**, а в случае использования для спецификации ГСА1 последнюю целесообразно преобразовать к одной из этих моделей графов переходов.

Это, естественно, не исключает применение графов переходов для других классов автоматов. Аналогичная классификация может быть предложена для графов переходов автоматов **Мили**, в которых значения выходных переменных формируются не в вершинах графов переходов, а на дугах. Во многих задачах логического управления переход от модели автомата **Мура** к модели автомата **Мили** не уменьшает числа состояний (вершин графа переходов). На рис.7 в качестве примера приведен граф переходов автомата **Мура**, реализующий счетный триггер, а на рис.8 эта же задача описана с помощью графа переходов автомата **Мили.**

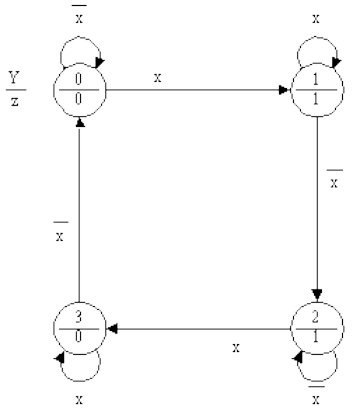


Рис. 7. Граф переходов автомата **Мура**

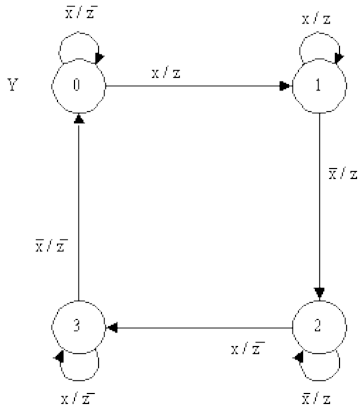


Рис. 8. Граф переходов автомата **Мили**

В других случаях переход от графа переходов автомата Мура или графа переходов автомата без выходного преобразователя к графу переходов автомата Мили позволяет сократить число вершин. На рис.9 приведен граф переходов автомата без выходного преобразователя с четырьмя вершинами, последовательностный одноразрядный сумматор, а на рис.10 этот алгоритм описан графом переходов автомата Мили с двумя вершинами.

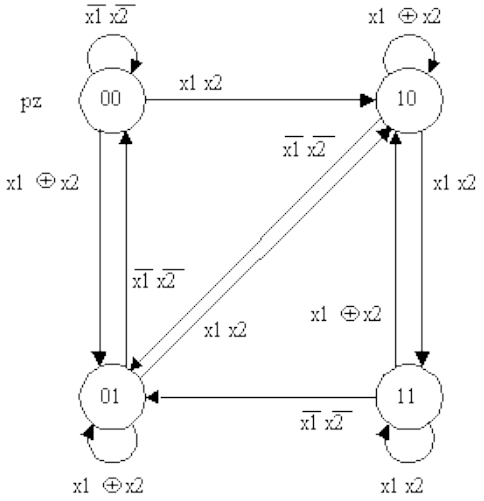


Рис. 9. Граф переходов автомата Мура Мили

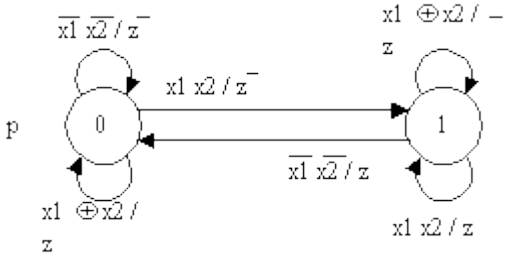


Рис. 10. Граф переходов автомата

При этом необходимо отметить, что для автоматов Мура и автоматов без выходного преобразователя с однозначным заданием значений выходов число состояний равно числу комбинаций этих значений (среди которых учитываются повторяющиеся). Для автоматов, в которых все комбинации различны, равно числу различных комбинаций. А для автоматов этих классов с неоднозначным их заданием понятие "состояние" становится менее связанным со значениями выходов и поэтому становится более абстрактным и менее понятным.

Это положение усугубляется для автоматов Мили, а для автоматов этого класса с неоднозначными значениями выходов вместо понятия "состояние" было введено понятие "ситуация", а вместо термина "граф переходов" - термин "граф переключений". Однако, несмотря на то, что граф

переключений может содержать меньшее число вершин, чем эквивалентный граф переходов, применение графов переключений в качестве языка общения нецелесообразно ввиду сложности их понимания.

3 Микропрограммные автоматы на ПЗУ

Микропрограммные автоматы представляют собой следующий шаг по пути усложнения интеллекта цифровых схем. На основе микропрограммных автоматов можно строить устройства, которые работают по довольно сложным алгоритмам, выполняют различные функции, определяемые входными сигналами, выдают сложные последовательности выходных сигналов.

В отличие от рассматривавшихся ранее устройств на "жесткой" логике, принцип работы которых однозначно определяется используемыми элементами и способом их соединения, микропрограммные автоматы с помощью одной и той же схемы могут выполнять самые разные функции. То есть они гораздо более гибкие, чем схемы на "жесткой" логике. К тому же проектировать микропрограммные автоматы с точки зрения схемотехники довольно просто. Недостатком любого микропрограммного автомата по сравнению со схемами на "жесткой" логике является меньшее предельное быстродействие и необходимость составления карты прошивки ПЗУ с микропрограммами, часто довольно сложными.

Наиболее распространенная структура микропрограммного автомата (включает в себя всего лишь три элемента: ПЗУ, регистр, срабатывающий по фронту, и тактовый генератор.

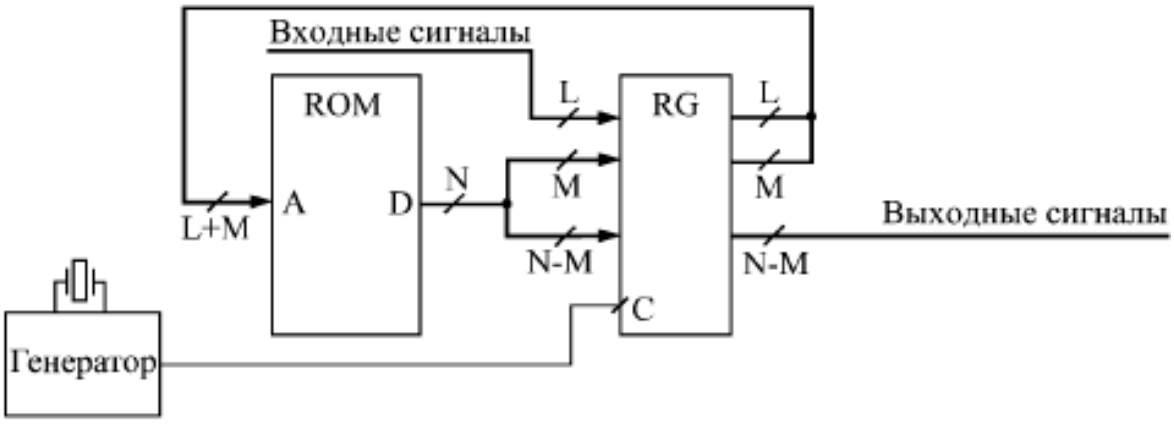


Рис. 11. Структура микропрограммного автомата

ПЗУ имеет (L+M) адресных разрядов и N разрядов данных. Регистр применяется с количеством разрядов (N + L). Разряды данных ПЗУ записываются в регистр по положительному фронту тактового сигнала с генератора. Часть этих разрядов (M) используется для образования адреса ПЗУ, другая часть (N-M) служит для формирования выходных сигналов. Входные сигналы (L) поступают на входы регистра и используются совместно с частью выходных разрядов ПЗУ для получения адреса ПЗУ.

Схема работает следующим образом. В каждом такте ПЗУ выдает код данных, тем самым, определяя не только состояние выходных сигналов

схемы, но и адрес ПЗУ, который установится в следующем такте (после следующего положительного фронта тактового сигнала). На этот следующий адрес влияют также и входные сигналы. То есть в отличие от формирователя последовательности сигналов, рассмотренного в предыдущем разделе, в данном случае адреса могут перебираться не только последовательно (с помощью счетчика), но и в произвольном порядке, который определяется прошивкой ПЗУ, называемой микропрограммой.

Условием правильной работы схемы будет следующее. За один период тактового сигнала должны успеть сработать регистр и ПЗУ. Иначе говоря, сумма задержки регистра и задержки выборки адреса ПЗУ не должна превышать периода тактового сигнала. Отметим также, что входные сигналы микропрограммного автомата нельзя подавать непосредственно на адресные входы ПЗУ (без регистра), так как их асинхронное (по отношению к тактовому сигналу) изменение может вызвать переходный процесс на выходах данных ПЗУ именно в тот момент, когда выходные сигналы ПЗУ записываются в регистр. В результате в регистр может записаться неверная информация, что нарушит работу всей схемы. Регистр же синхронизирует изменения входных сигналов с тактовым сигналом, в результате чего все разряды кода адреса ПЗУ меняется одновременно — по положительному фронту тактового сигнала. Регистр должен обязательно срабатывать по фронту, использование регистра-защелки не допускается, так как он может вызвать лавинообразный переходный процесс.

Возможности такой простой схемы оказываются очень большими. Например, микропрограммный автомат может выдавать последовательности выходных сигналов в ответ на определенное изменение входных сигналов. Он может также временно остановить выдачу выходных сигналов до прихода входных сигналов. Он может анализировать длительность входного сигнала и в зависимости от нее выдавать те или иные выходные сигналы. Он может также и многое другое.

Сформулируем несколько элементарных функций, из которых могут складываться алгоритмы работы микропрограммного автомата (рисунок 12).

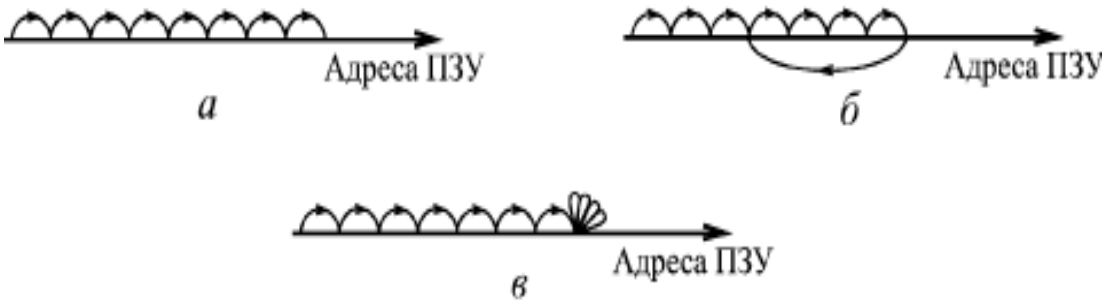


Рис. 12. Элементарные функции микропрограммного автомата: последовательный перебор (а), циклическое повторение (б) и остановка (в)

а) Последовательный перебор адресов ПЗУ (например, для выдачи последовательности выходных сигналов).

б) Периодическое повторение последовательности адресов ПЗУ (например, для повторения последовательности выходных сигналов).

в) Остановка в каком-то адресе ПЗУ (например, для ожидания изменения входного сигнала).

г) Временное отключение реакции на входные сигналы (например, для того, чтобы закончить отработку реакции на предыдущее изменение входных сигналов). Эта функция на рисунке не показана.

В качестве примера рассмотрим выполнение некоторых элементарных функций микропрограммным автоматом, показанным на рисунке 13.

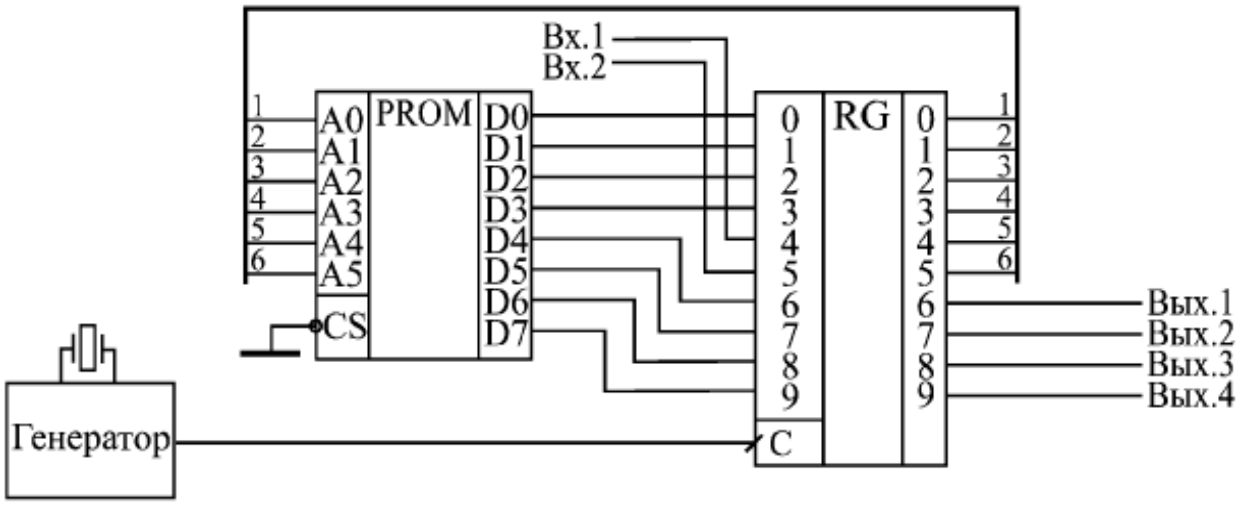


Рис. 13. Пример практической схемы микропрограммного автомата на ПЗУ

ПЗУ имеет организацию 64x8 (это могут быть две микросхемы РЕ3 или одна микросхема РТ18), количество разрядов регистра равно десяти (это могут быть две микросхемы ИР27). Схема имеет два входных сигнала и четыре выходных сигнала. Такая организация микропрограммного автомата хороша тем, что его микропрограмма (то есть карта прошивки ПЗУ) очень наглядна, легко составляется и читается.

Таблица 1. Карта прошивки ПЗУ 64x8 для микропрограммного автомата

Адрес	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
10	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
20	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
30	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

Выходные разряды данных ПЗУ делятся на две группы по 4 сигнала: младшие идут на образование следующего адреса ПЗУ, старшие образуют четыре выходных сигнала. Входные сигналы поступают (через регистр) на два старших разряда адреса ПЗУ. Если мы изобразим карту прошивки ПЗУ в

привычном для нас виде таблицы из четырех строк и 16 столбцов (табл. 1), то в этой таблице будут наглядно видны как состояние каждого из внутренних сигналов, так и реакция схемы на любой входной сигнал, а также состояния всех выходных сигналов в каждом такте.

Прежде всего, легко заметить, что выбор той или иной строки таблицы производится старшими разрядами кода адреса ПЗУ, то есть входными сигналами микропрограммного автомата. Таким образом, любое изменение входных сигналов приводит к переходу в карте прошивки на другую строку. Например, строка 00 будет соответствовать нулевым входным сигналам микропрограммного автомата, а строка 30 — единичным входным сигналам.

Таблица 2. Пример микропрограммы для схемы на рис. 11.17

Адрес	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	11	22	33	44	55	66	77	88	99	AA	BB	CC	DD	EE	FF	00
10	11	22	33	44	55	66	77	88	99	55	BB	CC	DD	EE	FF	00
20	10	21	32	43	54	65	76	87	98	A9	BA	CB	DC	ED	FE	0F
30	10	20	30	40	50	60	70	80	90	A0	B0	C0	D0	E0	F0	00

Теперь посмотрим на структуру 8-разрядного кода данных ПЗУ. Младшие четыре разряда этого кода (правый, младший знак 16-ричного кода в таблице) соответствуют четырем младшим разрядам кода адреса ПЗУ. Старшие четыре разряда кода данных ПЗУ (левый, старший знак 16-ричного кода в таблице) соответствуют четырем выходным сигналам микропрограммного автомата. То есть непосредственно по 16-ричному коду данных из таблицы можно сказать, во-первых, каким будет следующий адрес ПЗУ, и во-вторых, какими будут выходные сигналы автомата в следующем такте.

Рассмотрим пример микропрограммы, реализующей некоторые элементарные функции.

Верхняя (нулевая) строка таблицы демонстрирует последовательный перебор адресов памяти при нулевых входных сигналах. Пусть, например, автомат находится в адресе 00. В ячейке с адресом 00 указан следующий адрес 1 (младший знак 16-ричного кода 11), то есть в следующем такте автомат перейдет в адрес 01 (считаем, что входные сигналы остаются нулевыми). Из адреса 01 автомат перейдет в адрес 02, так как в ячейке с адресом 01 указан следующий адрес 2. Точно так же из адреса 02 автомат перейдет в адрес 03 и так далее до адреса 0F, в котором указан следующий адрес 0, то есть в следующем такте автомат снова вернется в адрес 00. Затем цикл последовательного прохождения адресов первой строки повторится (если, конечно, входные сигналы останутся нулевыми). Четыре выходных сигнала автомата в данном случае повторяют код следующего адреса, то есть, подобно 4-разрядному двоичному счетчику, выдают постепенно нарастающий код.

Вторая сверху (первая) строка таблицы демонстрирует циклическое повторение группы тактов. Если, например, работа начинается с адреса 10, то микропрограммный автомат последовательно будет перебирать адреса 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, а затем вернется в адрес 15 и будет постоянно повторять группу адресов 15, 16, 17, 18, 19. В данном случае мы считаем, что входной сигнал "Вх. 1" постоянно равен единице, а входной сигнал "Вх. 2" постоянно равен нулю, что и соответствует второй сверху строке таблицы.

Посмотрим теперь, что будет делать автомат, если входной сигнал "Вх. 2" постоянно равен нулю, а входной сигнал "Вх.1" меняет свое состояние с нуля на единицу и обратно. Такое переключение входных сигналов приводит к переходу между нулевой и первой строками таблицы с микропрограммой. Допустим, автомат начинает работу с адреса 00 (сигнал "Вх. 1" равен нулю). Идет последовательный перебор адресов нулевой строки. Даже если в течение первых девяти тактов сигнал "Вх. 1" будет меняться, на выходных сигналах автомата это никак не отразится, так как коды первых девяти адресов в нулевой и первой строках полностью совпадают. Поэтому можно сказать, что в этих первых девяти тактах мы отключили реакцию нашего автомата на входной сигнал "Вх. 1" за счет дублирования кодов.

Допустим теперь, что изначально нулевой входной сигнал "Вх. 1" стал равен единице в адресе 0A и далее не меняется. Это приведет к тому, что вместо адреса 0B автомат перейдет в адрес 1B. Затем он пройдет адреса 1C...1F, перейдет в адрес 10, дойдет до адреса 19 и начнет повторять цикл 15...19. Если же, например, в адресе 18 сигнал "Вх. 1" снова станет равен нулю, то вместо адреса 19 автомат попадет в адрес 09 и далее будет выполнять микропрограмму нулевой строки.

Третья сверху (вторая) строка таблицы, которая соответствует входным сигналам Вх. 1 = 0, Вх.2 = 1, показывает пример остановки автомата в каждом адресе и ожидание прихода входного сигнала. Например, автомат находится в адресе 23. Следующим адресом в коде данной ячейки указан 3. Значит, если входные сигналы остаются неизменными, то автомат перейдет опять же в адрес 23 и будет оставаться в нем постоянно. Но если входной сигнал "Вх.2" станет равным нулю, автомат перейдет в адрес 03 и начнет выполнять микропрограмму нулевой строки. То есть переключение входного сигнала "Вх.2" из нуля в единицу при нулевом уровне "Вх.1" останавливает выполнение микропрограммы нулевой строки до обратного перехода входного сигнала "Вх.2" в нуль.

Наконец нижняя (третья) строка таблицы демонстрирует переход из любого адреса строки в нулевой адрес этой же строки. Пусть, например, выполняется микропрограмма нулевой строки ("Вх. 1" и "Вх. 2" — нулевые) и в адресе 06 оба входных сигнала переходят в единицу. Автомат попадает в адрес 37, а из него — в адрес 30, где и остается постоянно до изменения входных сигналов. Если затем оба входных сигнала снова станут нулевыми, то автомат перейдет в адрес 00 и начнет снова выполнять микропрограмму нулевой строки таблицы.

Иногда бывает необходимо перевести автомат в какой-то определенный адрес. Ведь при включении питания в регистре может оказаться произвольный код. Для такой инициализации автомата можно, например, использовать регистр, имеющий вход сброса R, на который подается внешний сигнал, и тогда по этому сигналу автомат попадет в нулевой адрес. Но можно пойти и другим путем: составить микропрограмму таким образом, что автомат при включении питания сам перейдет за несколько тактов в нужный адрес независимо от того, в какой адрес он попал при включении питания.

Рассмотрим теперь пример проектирования простейшего микропрограммного автомата.

Пусть нам необходимо формировать с помощью микропрограммного автомата последовательность из трех выходных сигналов в ответ на положительный фронт одного входного сигнала.

Рис. 14. Диаграмма работы проектируемого микропрограммного автомата

Второй выходной сигнал "вложен" в первый, а третий "вложен" во второй. Причем до тех пор, пока данная последовательность не закончится, входной сигнал может произвольно менять свое состояние, на работе схемы это никак не должно сказываться. А после окончания данной последовательности микропрограммный автомат снова должен ждать положительного фронта входного сигнала и начинать по нему выработку новой последовательности.

Все временные сдвиги в выходной последовательности (t) примем для простоты одинаковыми и равными 1 мкс. Это значительно облегчает выбор тактовой частоты микропрограммного автомата, она должна быть равной 1 МГц (период 1 мкс). В этом случае полная длина генерируемой последовательности составит всего 6 тактов (один такт соответствует переходу всех выходных сигналов в единицу).

Для формирования 6-тактовой последовательности необходимо не менее 3 адресных разрядов ПЗУ (так как $2^2 = 4$, а $2^3 = 8$). Еще один разряд адреса ПЗУ потребуется для входного сигнала схемы. Итого необходимо 4 адресных разряда ПЗУ.

Литература: 5, стр. 68-79; 9, стр. 32-54; 2, стр.123-125; 3, стр.78-84

Контрольные вопросы

1 Что такое автоматы?
2 Что такое автоматы Мура?
3 Что такое автоматы Мили?