7. Синхронизация на уровне распознавания и обработки кадров и ячеек

Как уже отмечалось, передаваемые по транспортным системам данные объединяются в логически законченные структурные единицы: кадры, пакеты, ячейки и т. п. Поэтому приемник данных должен распознавать не только передаваемые биты, но и построенные из них кадры или иные информационные посылки. В этом смысле можно говорить о синхронизации приемника с передатчиком на уровне передачи таких посылок.

Обычно кадр начинается с флага – уникальной комбинации битов (например 01111110), которая не встречается в его оставшейся части. Чтобы достичь такой уникальности, оставшаяся часть кадра (область данных) перед выдачей в линию анализируется и при необходимости “разбавляется” вставкой служебных нулевых битов в имеющиеся длинные цепочки единиц (эта процедура называется битстаффингом). В данном примере флаг содержит шесть единиц; поэтому после каждых пяти единиц, следующих вплотную друг за другом в области данных, всегда (независимо от значения последующего бита) вставляется нулевой (служебный) бит. Тем самым исключается возможность последующего обнаружения удаленным устройством группы из шести или более следующих друг за другом единиц в области данных, что могло бы привести к ложному опознанию флага. Флаг при передаче не подвергается битстаффингу и поэтому сохраняет уникальность, что позволяет распознать его удаленным приемником.

После получения кадра, переданного по линии связи, и обнаружения начала области данных выполняется операция, обратная битстаффингу. Нулевой бит, следующий за каждой группой из пяти единиц в области данных, справедливо расценивается приемником как служебный и безусловно вычеркивается. Поэтому, в частности, длинные цепочки единиц смыкаются, полученное поле данных кадра обретает первоначальный вид. Из-за применения битстаффинга длина передаваемых по линии кадров непостоянна: например, кадр, содержащий в поле данных “сплошные нули”, передается по линии без изменения, а кадр со “сплошными единицами” в поле данных требует значительного разбавления нулями. (При постоянной длине кадра непостоянным окажется число размещенных в нем байтов данных.) Таким образом, избыточность потока передаваемых по линии данных обусловлена как применением многоразрядного флага, так и битстаффингом.

В начале этой главы (п. 7.1) рассмотрены схемы распознавания многоразрядных флагов в битовом и байтовом потоках данных. Далее (п. 7.2, 7.3) приведены схемы распознавания одноразрядных и раздробленных флагов, а также методы определения границ ячеек без использования флаговых битов (п. 7.4). В решениях по п. 7.5 – 7.7 учитываются возможные искажения данных из-за помех в линии. Способ передачи данных по п. 7.8 позволяет повысить эффективность использования избыточных битов синхронизации. В п. 7.9 рассмотрены решения задачи экономичного размещения низкоскоростного потока данных в высокоскоростном потоке кадров. В конце главы (п. 7.10) приведен способ уменьшения числа операций, связанных с распознаванием флагов начала кадров.

Начнем с относительно простых решений.

7.1. Распознавание многоразрядного флага в битовом и байтовом потоках данных

7.1.1. Распознавание флага в битовых потоках данных

Схема, приведенная на рис. 7.1, *а*, предназначена для распознавания девятиразрядного флагового кода (флага) во входном потоке данных. В данном примере флаг выбран равным 000000001. Он предварительно загружается в девятиразрядный регистр RG2.

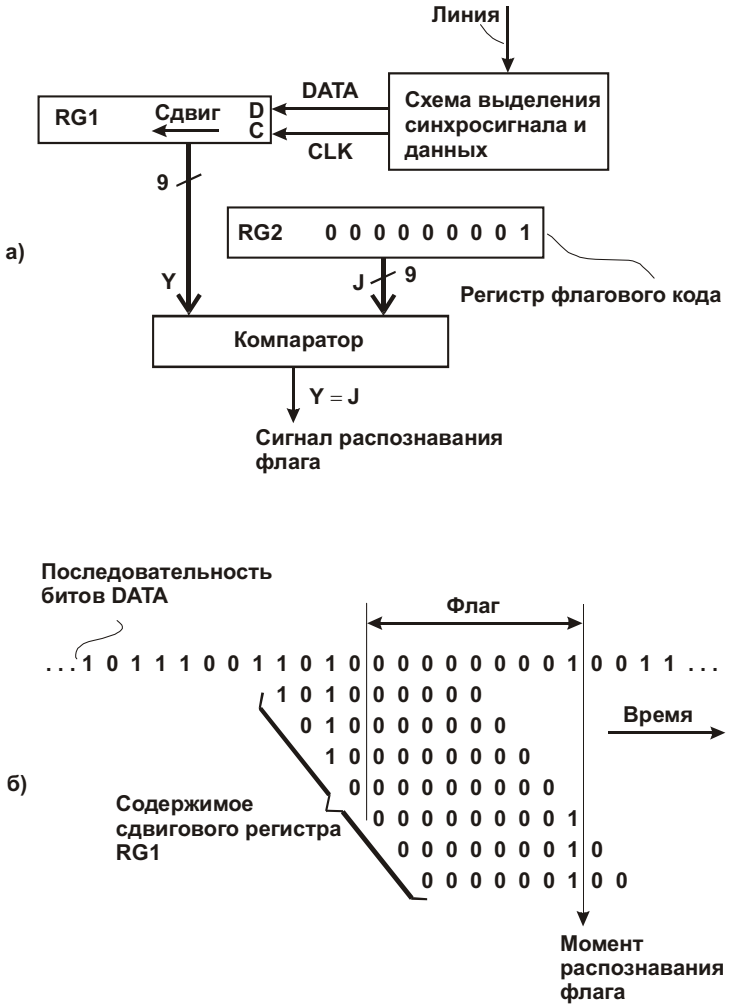

Рис. 7.1. Распознавание флага в битовом потоке данных: *а* – схема на основе многоразрядного компаратора; *б* – прохождение кода через сдвиговый регистр RG1

Из поступающего по линии сигнала выделяются синхросигнал CLK и данные DATA (код NRZ). Под действием синхросигнала данные загружаются в сдвиговый регистр RG1 и последовательно продвигаются в нем, как показано на рис. 7.1, *б*. При совпадении кодов в регистрах RG1 и RG2 срабатывает компаратор, на его выходе формируется сигнал распознавания флага. Длительность этого сигнала равна одному периоду (такту) синхросигнала CLK. В моменты изменения данных в регистре RG1 на выходе компаратора могут наблюдаться ложные срабатывания, поэтому для фиксации истинного значения выходного сигнала (Y = J) его следует принять на триггер (триггер на рисунке не показан). Синхронизация этого триггера может осуществляться задержанным (на время, превышающее задержку компаратора) сигналом CLK либо

непосредственно этим сигналом. В последнем случае результат сравнения кодов будет запаздывать на один такт.

Несмотря на простоту данного решения, его применимость ограничена сравнительно низкоскоростными системами передачи данных. Это связано с тем, что переходные процессы на выходе компаратора должны закончиться к началу очередного битового интервала (такта), длительность которого при высоких скоростях передачи данных может составлять, например, 3 нс или менее. С увеличением разрядности флагового кода число логических элементов компаратора возрастает, увеличивается его задержка; поэтому, чтобы уложиться в битовый интервал, необходимо применение все более быстродействующей элементной базы. А это нежелательно, так как повышается потребляемая мощность, уменьшается степень интеграции и т. д.

Чтобы снизить требования к быстродействию элементной базы, следует уменьшить число элементарных задержек на пути распространения сигнала в течение одного такта. Одним из широко применяемых методов уменьшения задержек (или, что то же самое – повышения максимальной тактовой частоты) является конвейерный метод обработки сигналов. Поясним его сущность на примере преобразования структуры рассмотренного ранее (рис. 7.1, *а*) компаратора к конвейерному виду. Предположим, что компаратор первоначально выполнен в виде пирамидальной комбинационной (не содержащей элементов памяти и цепей обратной связи) схемы, показанной на рис. 7.2, *а*. Суммарная задержка такого компаратора равна пяти задержкам логических элементов и должна быть меньше длительности такта.

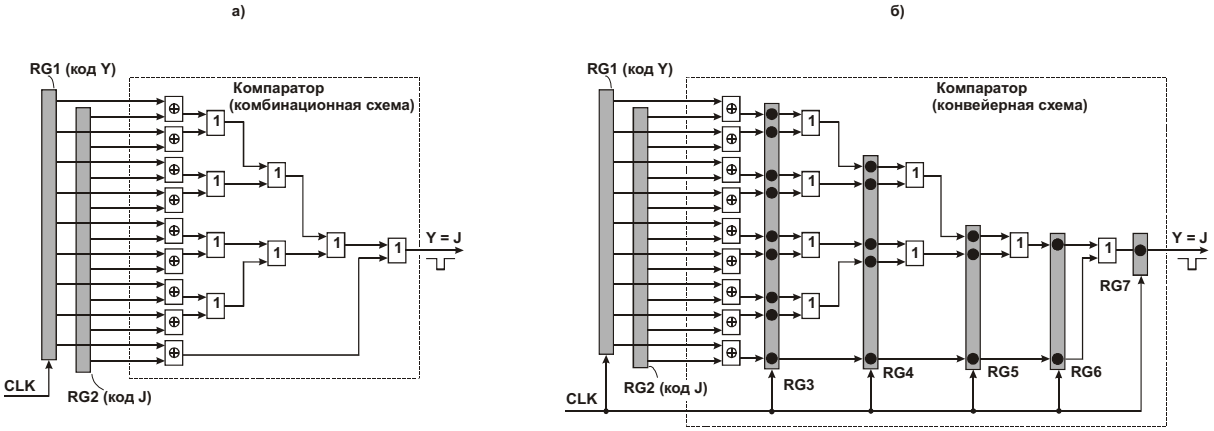

Рис. 7.2. Преобразование комбинационной схемы компаратора в конвейерную для повышения максимальной тактовой частоты сравнения кодов Y и J

Группа элементов Исключающее ИЛИ выполняет поразрядное сравнение кода Y в сдвиговом регистре RG1 с флаговым кодом в регистре RG2. При совпадении кодов Y и J на выходах этих элементов формируются сигналы лог. 0, которые проходят через пирамидальную структуру из элементов на выход компаратора. Если коды Y и J не совпадают, на выходе компаратора формируется сигнал лог. 1.

Для преобразования комбинационной схемы компаратора в конвейерную она делится на ярусы, разделенные регистрами (RG3 – RG7), в которых фиксируются промежуточные и окончательный результаты вычислений (рис. 7.2, *б*). В данном примере использовано максимально возможное расслоение комбинационной схемы, при котором каждый ярус имеет задержку, не превышающую задержку одного логического элемента. Регистры RG3 – RG7 содержат соответственно девять, пять, три, два и один D-триггер, обозначенные на рисунке точками. Регистры RG1, RG3 – RG7 синхронизируются положительными фронтами тактового сигнала CLK.

Предположим, что в начале такта T_0 в регистр RG1 (рис. 7.2, *б*) поступил очередной код Y0 для сравнения с флаговым кодом J, постоянно хранящемся в регистре

RG2. К моменту окончания такта T_0 (с некоторым запасом) на выходах элементов Исключающее ИЛИ сформирован результат поразрядного сравнения кодов Y0 и J. В начале следующего такта T_1 этот результат запоминается в регистре RG3. Одновременно с этим данные в регистре RG1 сдвигаются на один разряд, в результате в регистре RG1 формируется код Y1.

К моменту окончания такта T_1 на входах регистров RG4 и RG3 сформированы установившиеся сигналы, отображающие промежуточные результаты обработки кодов Y0 и Y1. Эти сигналы запоминаются в регистрах в начале такта T2, при этом в регистре RG1 формируется очередной код Y2 и т. д. Таким образом, в начале такта T_5 в регистре RG7 зафиксирован окончательный результат обработки кода Y0, а в регистрах RG6 – RG3 – промежуточные результаты обработки кодов Y1 – Y4. Иными словами, процесс обработки, как и при использовании "настоящего" конвейера, распадается на ряд простых и быстро выполняемых операций.

Рассмотренное решение (рис. 7.2, *б*) позволяет регистрировать точные совпадения кодов Y и J. Однако на практике точные совпадения могут наблюдаться не всегда из-за наличия ошибок в канале связи. Поэтому для исчерпывающей оценки результатов поиска необходимы интеллектуальные системы, проявляющие "терпимость" к ошибкам. В частности, аппаратура поиска должна не только давать ответ типа "да – нет", но и оценивать степень сходства сравниваемых кодов в случае их несовпадения.

Схема, представленная на рис. 7.3 [52], осуществляет конвейерную обработку битового потока данных DATA и в каждом такте формирует двоичное число Z3 – Z0 (Z0 – младший разряд), которое отражает степень близости текущего кода к восьмиразрядному флаговому коду.

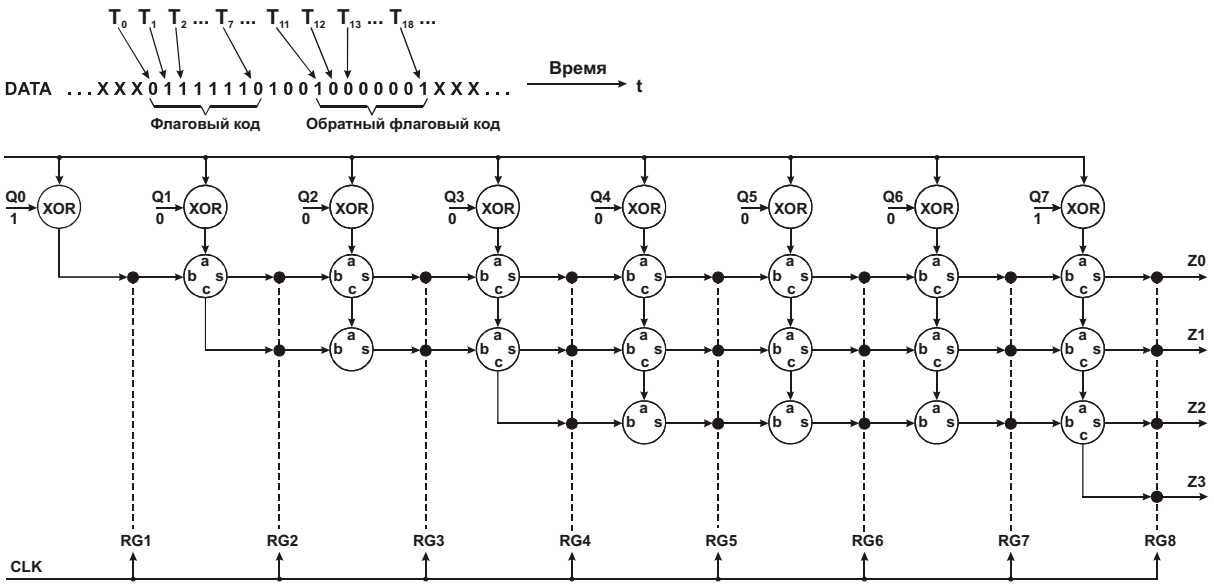

Рис. 7.3. Конвейерная схема распознавания флага в битовом потоке данных – первый вариант

Схема содержит восемь элементов Исключающее ИЛИ (XOR), семнадцать двухвходовых битовых сумматоров и восемь регистров RG1 – RG8. Разряды регистров выполнены на D-триггерах, условно показанных точками. Все триггеры синхронизируются положительными фронтами тактового сигнала CLK, сопровождающего биты данных DATA. Первые входы элементов Исключающее ИЛИ объединены и предназначены для приема последовательных данных DATA. На вторые входы этих элементов подан инвертированный флаговый код; прямой флаговый код равен 01111110_2. Сумматор выполняет арифметическую операцию сложения двух однобитовых чисел a и b: $s = a \oplus b$; $c = a \,\&\, b$, где s – сумма, c – перенос в следующий разряд.

Процесс прохождения через схему битовой последовательности, приведенной в верхней части рис. 7.3, поясняется Таблица 7.1. табл. 7.1. Символ “X” на рисунке и в таблице обозначает произвольное значение бита.

Таблица 7.1. табл. 7.1

Последовательность состояний конвейерной схемы (рис. 7.3)

t	DATA	Коды на входах регистров							
		RG1	RG2	RG3	RG4	RG5	RG6	RG7	RG8
T_0	0	1	X	X	X	X	X	X	X
T_1	1	0	2	X	X	X	X	X	X
T_2	1	0	1	3	X	X	X	X	X
T_3	1	0	1	2	4	X	X	X	X
T_4	1	0	1	2	3	5	X	X	X
T_5	1	0	1	2	3	4	6	X	X
T_6	1	0	1	2	3	4	5	7	X
T_7	0	1	0	1	2	3	4	5	8
T_8	1	0	2	1	2	3	4	5	5
T_9	0	1	0	2	1	2	3	4	6
T_{10}	0	1	1	0	2	1	2	3	5
T_{11}	1	0	2	2	1	3	2	3	3
T_{12}	0	1	0	2	2	1	3	2	4
T_{13}	0	1	1	0	2	2	1	3	3
T_{14}	0	1	1	1	0	2	2	1	4
T_{15}	0	1	1	1	1	0	2	2	2
T_{16}	0	1	1	1	1	1	0	2	3
T_{17}	0	1	1	1	1	1	1	0	3
T_{18}	1	0	2	2	2	2	2	2	0
T_{19}	X	X	X	X	X	X	X	X	X
T_{20}	X	X	X	X	X	X	X	X	X
T_{21}	X	X	X	X	X	X	X	X	X

Предположим, что в тактах T0 – T7 на вход конвейерной схемы поступает последовательность битов 01111110, совпадающая с флаговой. В результате ее обработки, как показано в таблице, в такте T7 на входе регистра RG8 формируется число 8, соответствующее последовательному восьмикратному совпадению поступивших битов DATA с соответствующими битами флагового кода. Этот результат получен следующим образом.

В такте T_0 DATA = 0, сигнал Q0 постоянно равен лог. 1, поэтому на входе одноразрядного регистра RG1 сформирован сигнал лог. 1 ($0 \oplus 1 = 1$). Так как исходное состояние регистров RG1 – RG8 не определено, то значения сигналов на входах реги-

стров RG2 – RG8 могут быть произвольными (см. символы “X” в первой строке таблицы).

В такте T_1 (вторая строка таблицы) DATA = 1, на входе регистра RG1 сформирован сигнал лог. 0 ($1 \oplus 1 = 0$). На вход регистра RG2 поступает число 2, так как на входы левого (по схеме) сумматора поданы две единицы. Первая из них поступает из регистра RG1 как результат вычисления, выполненного в предыдущем такте, вторая получена суммированием по модулю два сигналов DATA = 1 и Q1 = 0. Состояния сигналов на входах регистров RG3 – RG8 по-прежнему не определены.

В такте T_2 (третья строка таблицы) DATA = 1, на входах регистров RG1 – RG3 сформированы числа 0, 1 и 3. Из структуры данных в таблице видно, что с течением времени число неопределенных состояний (“X”) кодов в регистрах уменьшается, при этом они замещаются последовательно возрастающими числами 1, 2, 3 и т. д., размещенными по диагонали таблицы. Такая закономерность формирования чисел связана с тем, что в данном примере на вход схемы поступает код, совпадающий с флаговым. Поэтому амплитуда фронта проходящей по конвейеру “волны вычислений” неуклонно растет.

В конце такта T_7 на входе регистра RG8 сформирован код 8, который запоминается в этом регистре в начале такта T8. Как показано в правом столбце таблицы, в последующих тактах в регистре RG8 формируются коды 5, 6, 5, 3, 4, 3, 4, 2, 3, 3, 0. Последний (нулевой) код соответствует полному отсутствию совпадений битов кода DATA с битами флагового кода в период $T_{11} - T_{18}$.

Чтобы убедиться в правильности полученной последовательности кодов в регистре RG8 и пояснить их происхождение, рассмотрим модель вычислений, приведенную на рис. 7.4.

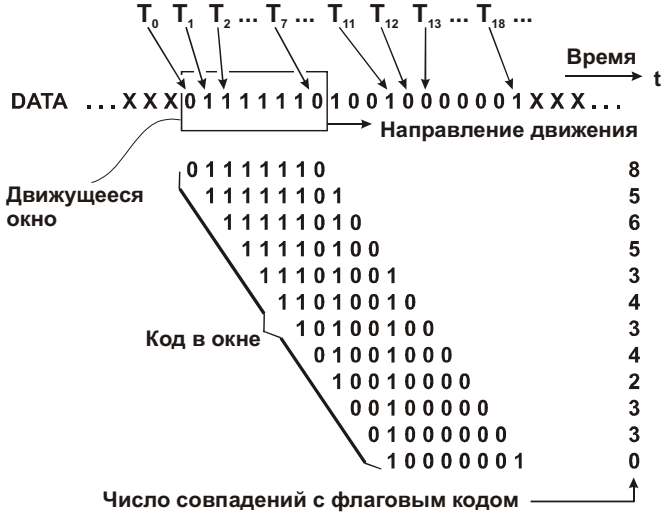

Рис. 7.4. Модель вычислений, поясняющая процесс конвейерной обработки кода DATA (см. рис. 7.3)

В данной модели конвейерной обработки кода DATA этот код просматривается через движущееся восьмиразрядное окно. Если окно зафиксировано в показанном на рисунке положении, то сквозь него просматривается код 01111110, совпадающий с флаговым, и, следовательно, совпадающий с ним восьми разрядах. В следующем такте окно перемещается на один разряд вправо, поэтому через него виден код 11111101. Этот код совпадает с флаговым в пяти разрядах. Как видно из рисунка, в последующих тактах числа (6, 5, 3, …), отражающие совпадения, полностью соответствует числам, приведенным в правом столбце таблицы.

Недостатком рассмотренной схемы (см. рис. 7.3) является сравнительно невысокая тактовая частота ее работы. Действительно, длительность такта (время между двумя положительными фронтами сигнала CLK) должна быть достаточной для надежного

срабатывания элемента Исключающее ИЛИ и трех последовательно включенных сумматоров (см. схему формирования сигналов на входе регистра RG8). Для устранения этого недостатка в [52] предлагается использовать идею “отложенных переносов”, которая заключается в том, что необязательно обрабатывать перенос в момент его возникновения – можно запомнить перенос и учесть его в последующих тактах суммирования. С учетом этой идеи рассмотренную схему можно преобразовать в более быстродействующую (рис. 7.5).

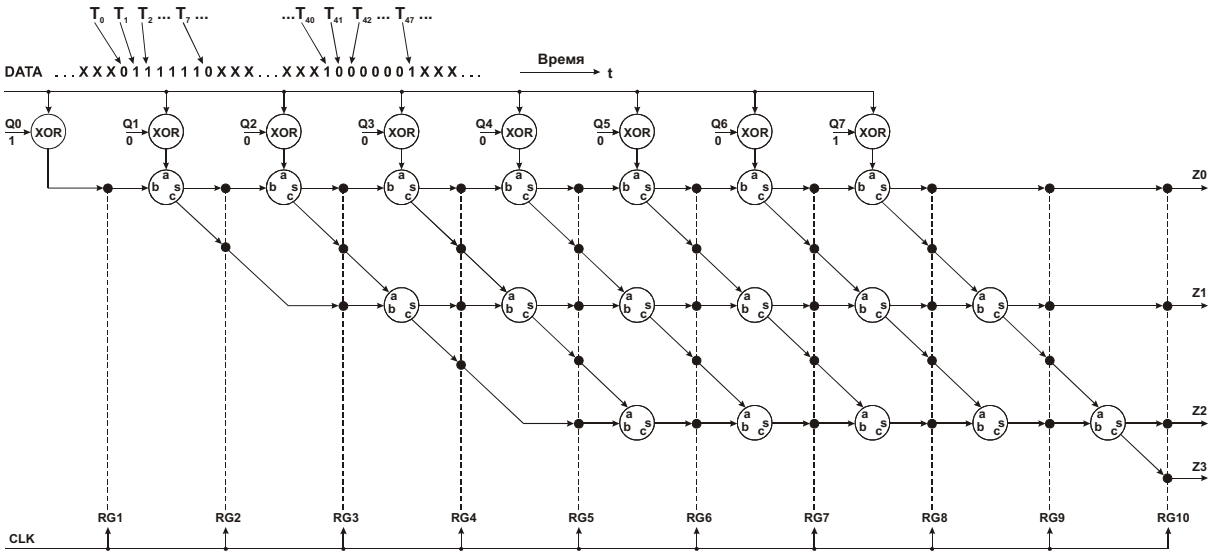

Рис. 7.5. Конвейерная схема распознавания флага в битовом потоке даных – второй вариант

В этой схеме максимальная задержка комбинационной схемы, включенной между двумя соседними регистрами, определяется задержкой цепи из элемента Исключающее ИЛИ и сумматора (по сравнению с предыдущей схемой исключены две задержки сумматора).

Так же как и при анализе состояний предыдущей схемы, воспользуемся табличным представлением сигналов на входах регистров (Таблица 7.2 табл. 7.2). В этой таблице, в отличие от предыдущей, коды представлены на уровне битов. Символами “-” обозначены отсутствующие разряды регистров. Символ “X” обозначает произвольное значение бита, символ “J” обозначает известное значение бита, которое, однако, не приводится как несущественное для описания идеи.

Таблица 7.2 табл. 7.2

Последовательность состояний конвейерной схемы (рис. 7.5)

t	DATA	Биты на входах регистров										Код на входе регистра RG10
		RG1	RG2	RG3	RG4	RG5	RG6	RG7	RG8	RG9	RG10	
T_0	0	1	X	X	X	X	X	X	X	X	X	X
		-	X	X	X	X	X	X	X	-	-	
		-	-	X	X	X	X	X	X	X	X	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_1	1	J	0	X	X	X	X	X	X	X	X	X
		-	1	X	X	X	X	X	X	-	-	
		-	-	X	X	X	X	X	X	X	X	

		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_2	1	J	J	1	X	X	X	X	X	X	X	X
		-	J	0	X	X	X	X	X	-	-	
		-	-	1	X	X	X	X	X	X	X	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_3	1	J	J	J	0	X	X	X	X	X	X	X
		-	J	J	1	X	X	X	X	-	-	
		-	-	J	1	X	X	X	X	X	X	
		-	-	-	0	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_4	1	J	J	J	J	1	X	X	X	X	X	X
		-	J	J	J	0	X	X	X	-	-	
		-	-	J	J	0	X	X	X	X	X	
		-	-	-	J	1	X	X	X	X	-	
		-	-	-	-	0	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_5	1	J	J	J	J	J	0	X	X	X	X	X
		-	J	J	J	J	1	X	X	-	-	
		-	-	J	J	J	0	X	X	X	X	
		-	-	-	J	J	0	X	X	X	-	
		-	-	-	-	J	1	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_6	1	J	J	J	J	J	J	1	X	X	X	X
		-	J	J	J	J	J	0	X	-	-	
		-	-	J	J	J	J	1	X	X	X	
		-	-	-	J	J	J	0	X	X	-	
		-	-	-	-	J	J	1	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_7	0	J	J	J	J	J	J	J	0	X	X	X
		-	J	J	J	J	J	J	1	-	-	
		-	-	J	J	J	J	J	1	X	X	
		-	-	-	J	J	J	J	0	X	-	
		-	-	-	-	J	J	J	1	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_8	X	X	X	X	X	X	X	X	X	0	X	X
		-	X	X	X	X	X	X	X	-	-	
		-	-	J	J	J	J	J	J	0	X	
		-	-	-	J	J	J	J	J	1	-	
		-	-	-	-	J	J	J	J	1	X	
		-	-	-	-	-	-	-	-	-	X	
T_9	X	X	X	X	X	X	X	X	X	X	0	8
		-	X	X	X	X	X	X	X	X	-	
		-	-	X	X	X	X	X	X	X	0	

		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	J	J	J	J	J	0	
		-	-	-	-	-	-	-	-	-	1	
T_{40}	1	0	X	X	X	X	X	X	X	X	X	X
		-	X	X	X	X	X	X	X	-	-	
		-	-	X	X	X	X	X	X	X	X	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{41}	0	J	0	X	X	X	X	X	X	X	X	X
		-	0	X	X	X	X	X	X	-	-	
		-	-	X	X	X	X	X	X	X	X	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{42}	0	J	J	0	X	X	X	X	X	X	X	X
		-	J	0	X	X	X	X	X	-	-	
		-	-	0	X	X	X	X	X	X	X	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{43}	0	J	J	J	0	X	X	X	X	X	X	X
		-	J	J	0	X	X	X	X	-	-	
		-	-	J	0	X	X	X	X	X	X	
		-	-	-	0	X	X	X	X	X	-	
		-	-	-	-	X	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{44}	0	J	J	J	J	0	X	X	X	X	X	X
		-	J	J	J	0	X	X	X	-	-	
		-	-	J	J	0	X	X	X	X	X	
		-	-	-	J	0	X	X	X	X	-	
		-	-	-	-	0	X	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{45}	0	J	J	J	J	J	0	X	X	X	X	X
		-	J	J	J	J	0	X	X	-	-	
		-	-	J	J	J	0	X	X	X	X	
		-	-	-	J	J	0	X	X	X	-	
		-	-	-	-	J	0	X	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{46}	0	J	J	J	J	J	J	0	X	X	X	X
		-	J	J	J	J	J	0	X	-	-	
		-	-	J	J	J	J	0	X	X	X	
		-	-	-	J	J	J	0	X	X	-	
		-	-	-	-	J	J	0	X	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{47}	1	J	J	J	J	J	J	J	0	X	X	X
		-	J	J	J	J	J	J	0	-	-	
		-	-	J	J	J	J	J	0	X	X	

		-	-	-	J	J	J	J	0	X	-	
		-	-	-	-	J	J	J	0	X	X	
		-	-	-	-	-	-	-	-	-	X	
T_{48}	X	X	X	X	X	X	X	X	X	0	X	X
		-	X	X	X	X	X	X	X	-	-	
		-	-	J	J	J	J	J	J	0	X	
		-	-	-	J	J	J	J	J	0	-	
		-	-	-	-	J	J	J	J	0	X	
		-	-	-	-	-	-	-	-	-	X	
T_{49}	X	X	X	X	X	X	X	X	X	X	0	0
		-	X	X	X	X	X	X	X	-	-	
		-	-	X	X	X	X	X	X	X	0	
		-	-	-	X	X	X	X	X	X	-	
		-	-	-	-	J	J	J	J	J	0	
		-	-	-	-	-	-	-	-	-	0	

Процессы, протекающие в схеме (рис. 7.5), в целом аналогичны рассмотренным при описании предыдущей схемы (рис. 7.3). Основное отличие связано с упоминавшейся ранее “отложенной” обработкой переносов из разряда в разряд, что увеличивает длину конвейера, но уменьшает время выполнения медленных операций. Из таблицы следует, что результат распознавания флага формируется на входе выходного регистра в такте T_9, а не в такте T_7, как в предыдущей схеме.

Отметим, что задержку комбинационных схем, включенных между регистрами, можно дополнительно уменьшить включением дополнительного восьмиразрядного регистра (с синхронизацией от сигнала CLK) в разрыв цепей между выходами элементов Исключающее ИЛИ и входами последующих элементов (D-триггера регистра RG1 и сумматоров). В этом случае задержки элементов Исключающее ИЛИ не будут складываться с задержками сумматоров. Можно пойти в этом направлении и далее – ввести регистры внутрь структур элементов Исключающее ИЛИ и сумматоров, чтобы свести задержку межрегистровой комбинационной схемы к задержке одного простейшего логического элемента типа И, ИЛИ, НЕ.

Рассмотрим некоторые модификации конвейерных схем. В схеме, приведенной на рис. 7.6, данные DATA поступают от приемника по двум каналам (а не по одному, как в предыдущих решениях). Такая организация потока данных свойственна, например, приемнику линейного сигнала с кодировкой 2B1Q. Правила кодирования следующие. Каждая пара выдаваемых в линию битов (“2B” – two binary) преобразуется в один из четырех (“1Q” – one of quaternary) уровней напряжения между проводами линии. В осциллограмме линейного сигнала при передаче случайных данных просматриваются четыре фиксированных уровня напряжения со случайными переходами между ними на границах между тактами. Приемник в каждом такте анализирует уровень сигнала и преобразует его в пару битов, которая одновременно выдается по двум каналам: DATA0 и DATA1.

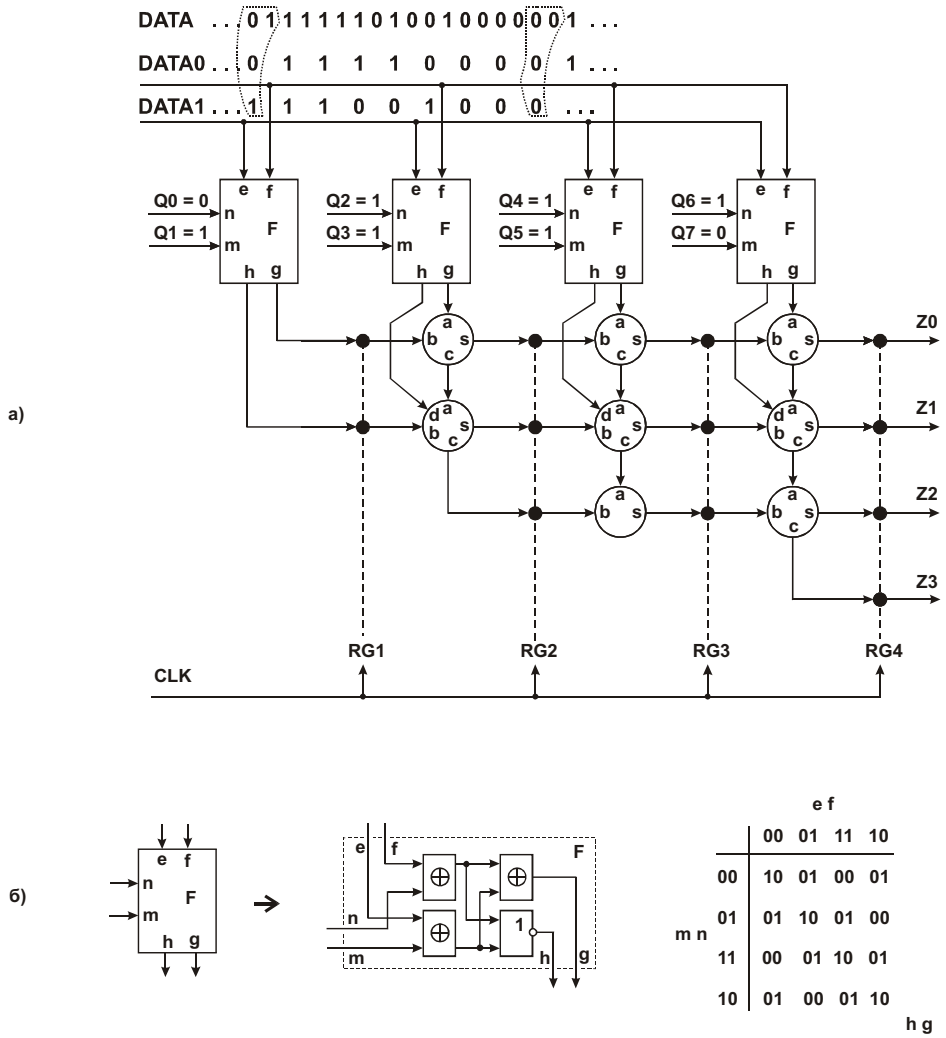

Рис. 7.6. Распознавание восьмибитового флага в битовом потоке даных: *а* – конвейерная схема – третий вариант; *б* – структура логического блока F и его таблица истинности

Данные из каналов DATA0 и DATA1 сравниваются логическими блоками F с соответствующими битами известного флагового кода. При одновременном совпадении бита f с битом n и бита e с битом m логический блок формирует двоичное число 2 ($hg = 10_2$). При неполном совпадении логический блок формирует число 1 ($hg = 01_2$). В отсутствие совпадений формируется нулевой код ($hg = 00_2$). Таким образом, логический блок осуществляет предварительную оценку степени совпадения кодов, которая далее учитывается при накоплении результирующей суммы, как уже было показано при описании предыдущих схем. В схеме (рис. 7.6) вместо двухвходовых применены три трехвходовых сумматора, выполняющих следующие функции: $s = a \oplus b \oplus d$; $c = a \,\&\, b + b \,\&\, d + a \,\&\, d$, где знак "+" соответствует логической операции ИЛИ, s – сумма, c – перенос в следующий разряд. Максимальное число в регистре RG4 равно восьми и соответствует опознанию флагового кода.

Схему, представленную на рис. 7.6 (и последующую, рис. 7.7), можно усовершенствовать для повышения тактовой частоты ее работы подобно тому, как это было сделано ранее (см. рис. 7.5), однако соответствующие решения здесь не приводятся.

Другая модификация конвейерной схемы (рис. 7.7) [52] позволяет работать с приемниками, имеющими встроенную аппаратуру контроля качества принятого сигнала. Если приемник гарантирует достоверность бита, выделенного из линейного сигнала, то комбинация сигналов W1W2 = 00 соответствует сигналу лог. 0, а комбинация W1W2 = 11 – сигналу лог. 1. Комбинации W1W2 = 01 и W1W2 = 10 формируются приемни-

ком при обнаружении ошибок, не позволяющих уверенно определить значение принятого бита.

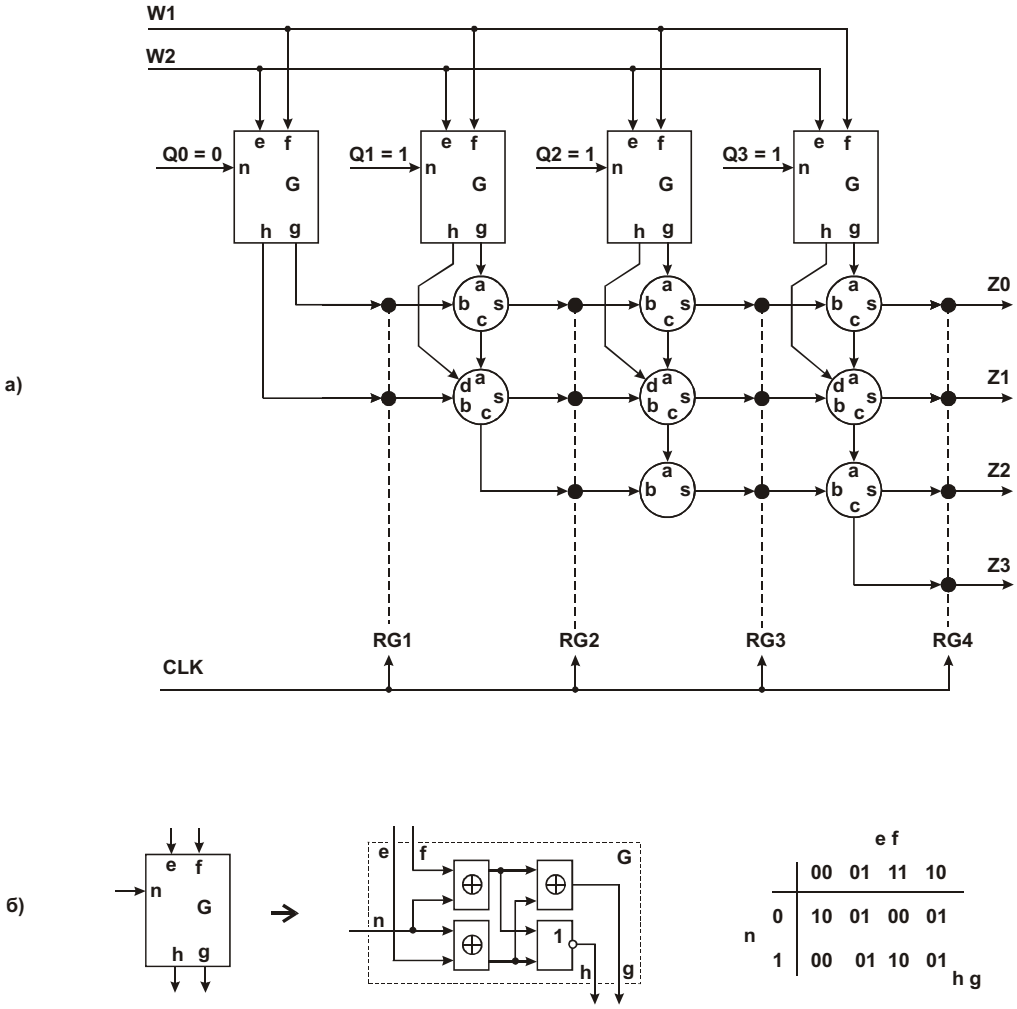

рис. 7.7. Распознавание четырехразрядного флага в битовом потоке даных: а – конвейерная схема – четвертый вариант; б – структура логического блока G и его таблица истинности

Блок G формирует число 2 ($hg = 10_2$) при наличии сигнала, уверенно распознанного приемником, если этот сигнал логически совпадает с поданным на вход n. Если уверенно распознанный приемником сигнал не совпадает с сигналом, поданным на вход n, то на выходах блока G формируется нулевой код ($hg = 00_2$). Неуверенная работа приемника сопровождается выдачей на выходы логического блока числа 1 ($hg = 01_2$).

После предварительного анализа данные с выходов логических блоков G обрабатываются точно также, как и в предыдущих схемах. Результат обработки – поток четырехразрядных чисел на выходе регистра RG4 – оценивается последующими устройствами. При уверенном приеме сигнала и точном совпадении принятого кода с четырехразрядным флаговым кодом (0111) в регистре RG4 в определенный момент возникает число 8. Неуверенный прием или (и) несовпадение принятых битов с флаговыми приводят к уменьшению этого числа.

С другими решениями задачи распознавания заданных кодов или их последовательностей в проходящем потоке данных Вы можете ознакомиться в [71, 72].

7.1.2. Распознавание флага в байтовых потоках данных

В высокоскоростных системах передачи данных принятый последовательный код сразу же преобразуется в параллельный, чаще всего восьмиразрядный. Дальнейшая обработка данных проводится на уровне байтов (а не битов), что в восемь раз снижает требуемую тактовую частоту. А это, в свою очередь, позволяет уменьшить потребляемую устройством мощность, снижает требования к элементной базе, размещению проводников на печатной плате и т. д.

В схеме, приведенной на рис. 7.8, также как и в ранее рассмотренной (см. рис. 7.1, *а*), из поступающего по линии сигнала выделяются синхросигнал CLK и данные DATA [55]. Под действием синхросигнала данные непрерывно загружаются в сдвиговый регистр RG1 и последовательно продвигаются в нем в направлении, указанном стрелкой. Выдвинутые за пределы регистра данные теряются.

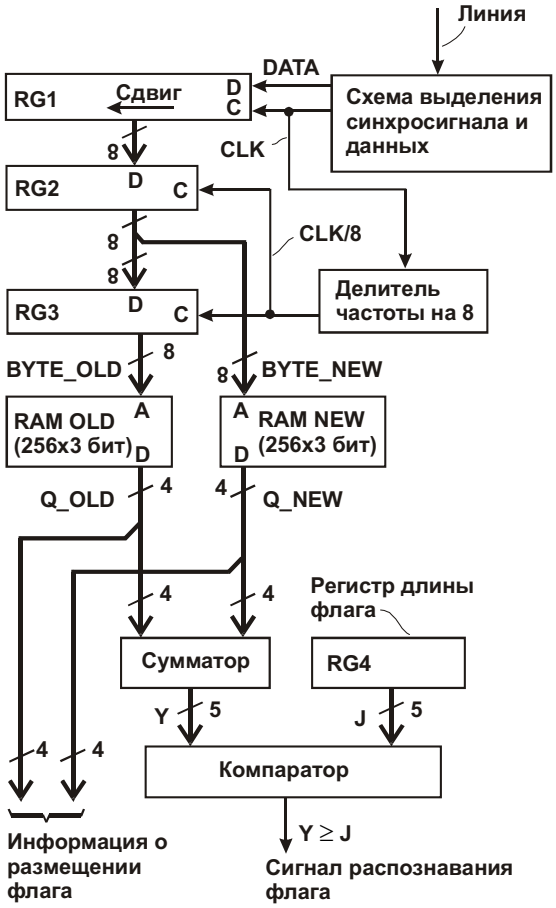

Рис. 7.8. Распознавание флага в байтовом потоке данных

Сигнал CLK проходит через делитель частоты на восемь (трехразрядный двоичный счетчик) и в виде сигнала CLK/8 поступает на входы синхронизации последовательно включенных параллельных регистров RG2 и RG3. Под действием этого сигнала в каждом восьмом такте сигнала CLK восьмиразрядный код из регистра RG2 переписывается в регистр RG3; одновременно с этим код из регистра RG1 переписывается в регистр RG2. Таким образом, битовая последовательность данных DATA преобразуется в поток байтов, который проходит по цепи RG2 – RG3. Начальное состояние делителя частоты не определено, поэтому разметка исходной битовой последовательности на байты произвольна – истинные границы байтов (которые подразумевались при выдаче данных удаленным передатчиком) в общем случае не совпадают с границами байтов в регистрах RG2 и RG3. Байты BYTE_NEW (новый байт) и BYTE_OLD (старый байт) в регистрах RG2 и RG3 образуют “наблюдаемое” в данный момент 16-

разрядное слово, в котором, возможно, содержится искомый девятиразрядный флаг. В момент очередного продвижения потока байтов (при поступлении сигнала CLK/8) наблюдаемое 16-разрядное слово смещается на один байт в сторону новых данных.

В каждом такте сигнала CLK/8 байты BYTE_OLD и BYTE_NEW анализируются для обнаружения в них левого и правого фрагментов флага (000000001). Для анализа используются блоки памяти RAM_OLD и RAM_NEW с соответствующими кодировками. Байты BYTE_OLD и BYTE_NEW используются для адресации ячеек этих блоков. Число предполагаемых совпадений соответствующих байтов с фрагментами флага отображается четырехразрядными кодами данных Q_OLD и Q_NEW, которые считываются из блоков памяти. Каждый из этих кодов принадлежит диапазону 0 – 8. Эти коды арифметически суммируются, в результате формируется двоичное число Y, по которому можно судить о наличии (или отсутствии) флага в паре регистров RG2 – RG3.

Чтобы получить окончательный результат анализа, число Y сравнивается с числом J (отображающим разрядность флага), предварительно загруженным в регистр длины флага. В данном примере использован девятиразрядный флаг, поэтому J = 9. Сигнал распознавания флага формируется при $Y \geq J$. При этом информация о размещении флага в наблюдаемом 16-разрядном слове (BYTE_OLD – BYTE_NEW) содержится в коде Q_OLD или (и) Q_NEW (в данном примере – в последнем).

Далее рассмотрены подробности этого решения.

Как показано в верхней части рис. 7.9, возможны восемь вариантов размещения флага в байтах BYTE_OLD – BYTE_NEW. Неоднозначность размещения связана с тем, что начальное состояние делителя частоты сигнала CLK не определено, и “нарезка” битовой последовательности на байты может начаться с произвольного места. В нижней части рисунка показаны схемы вычисления переменных Q_OLD и Q_NEW. Эти переменные отражают предполагаемое число битовых совпадений кодов с левым и правым фрагментами флага. Из верхней строки схемы вычисления переменной Q_OLD следует, что нулевой код в байте BYTE_OLD может (но не обязательно должен) соответствовать обнаружению восьми нулевых битов флага, поэтому Q_OLD = 8. Следующая строка показывает возможность совпадения кода с флагом в семи разрядах и т. д. Аналогичные рассуждения применимы и к схеме вычисления переменной Q_NEW.

Условие распознавания флага, как уже отмечалось, состоит в том, что

$$Y = Q\_OLD + Q\_NEW \geq 9.$$

Чтобы убедиться в его правильности, рассмотрим все благоприятные (соответствующие обнаружению флага) сочетания фрагментов кодов, приведенных в нижней части рис. 7.9. Эти сочетания обозначим двузначными числами, в которых первое соответствует коду Q_OLD, а второе – коду Q_NEW. Так, например, число 88 соответствует сочетанию фрагментов кодов 0000 0000 (BYTE_OLD) и 0000 0001 (BYTE_NEW). Полный 16-разрядный код равен 0000 0000 0000 0001. В правой части этого кода присутствует искомый девятиразрядный флаг 000000001. Его местоположение однозначно определяется кодом Q_NEW = 8. Еще один пример: числу 37 соответствует 16-разрядный код XXXX 1000 0000 001X, в котором содержится искомый флаг, и т. д. (Символы “X” соответствуют произвольным значениям битов.) Итак, благоприятные сочетания следующие:

88, 87, 86, 85, 84, 83, 82, 81,
78, 77, 76, 75, 74, 73, 72,
68, 67, 66, 65, 64, 63,
58, 57, 56, 55, 54,
48, 47, 46, 45,
38, 37, 36,
28, 27,
18.

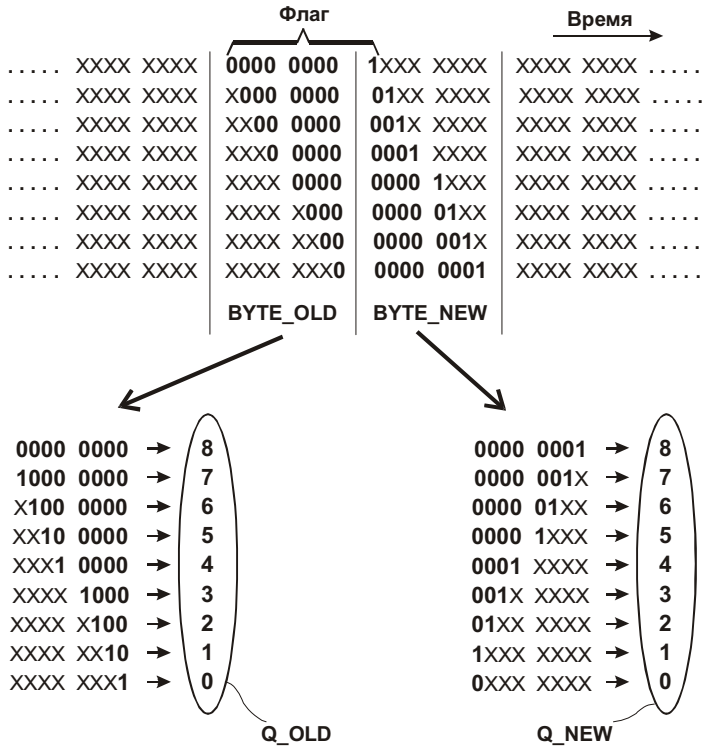

Рис. 7.9. Варианты размещения флага в байтах BYTE_OLD – BYTE_NEW и схемы вычисления переменных Q_OLD и Q_NEW. Символы "X" соответствуют произвольным значениям битов

Все эти сочетания характеризуются тем, что в каждом из них сумма первой и второй цифр превышает 8. Оставшиеся (неблагоприятные) сочетания в лучшем случае обеспечивают не более восьми совпадений с девятиразрядным флагом, что принципиально недостаточно для его распознавания.

Кодировка блоков памяти RAM_OLD и RAM_NEW представлена на рис. 7.10. Каждая ячейка блока памяти RAM_OLD (RAM_NEW) хранит четырехразрядное число Q_OLD (Q_NEW). Ячейки представлены на рисунке в виде массива чисел, принадлежащих диапазону от 0 до 8. Массив представлен матрицей из восьми строк и 32 столбцов (всего 256 ячеек). Адресация ячеек ведется слева направо, сверху вниз. Левая верхняя ячейка имеет адрес 00_{16}, правая нижняя – FF_{16}.

Поясним процесс распознавания флага на примере. Предположим, что после преобразования последовательного кода в параллельный в регистрах RG3 и RG2 сформированы коды BYTE_OLD = CO_{16} и BYTE_NEW = 10_{16}:

..... XXXX XXXX **1100 0000** **0001 0000** XXXX XXXX .....

$C0_{16}$ **10_{16}**

Как видим, в 16-разрядном коде присутствует флаг (000000001); пять его нулевых битов размещены в байте BYTE_OLD, а оставшиеся четыре бита (0001) – в байте BYTE_NEW. По адресу $C0_{16}$ из блока памяти RAM_OLD извлекается код 6. Одновременно с этим по адресу 10_{16} из блока памяти RAM_OLD извлекается код 4 (эти коды выделены на рис. 7.10 рамками). Сумма считанных из памяти кодов равна 10 (превышает длину флагового кода), поэтому на выходе компаратора формируется признак опознания флага. Положение правой границы флага однозначно определяется по приведенной в нижней части рис. 7.9 схеме вычисления кода Q_NEW. Действительно, зная результат вычисления (Q_NEW = 4) и продвигаясь от этого результата к породившим его условиям, можно утверждать, что правая граница флага неизбежно совпадает с

границей, разделяющей байт BYTE_NEW на две равные части (так как иное размещение границы привело бы к другому результату вычисления кода BYTE_NEW).

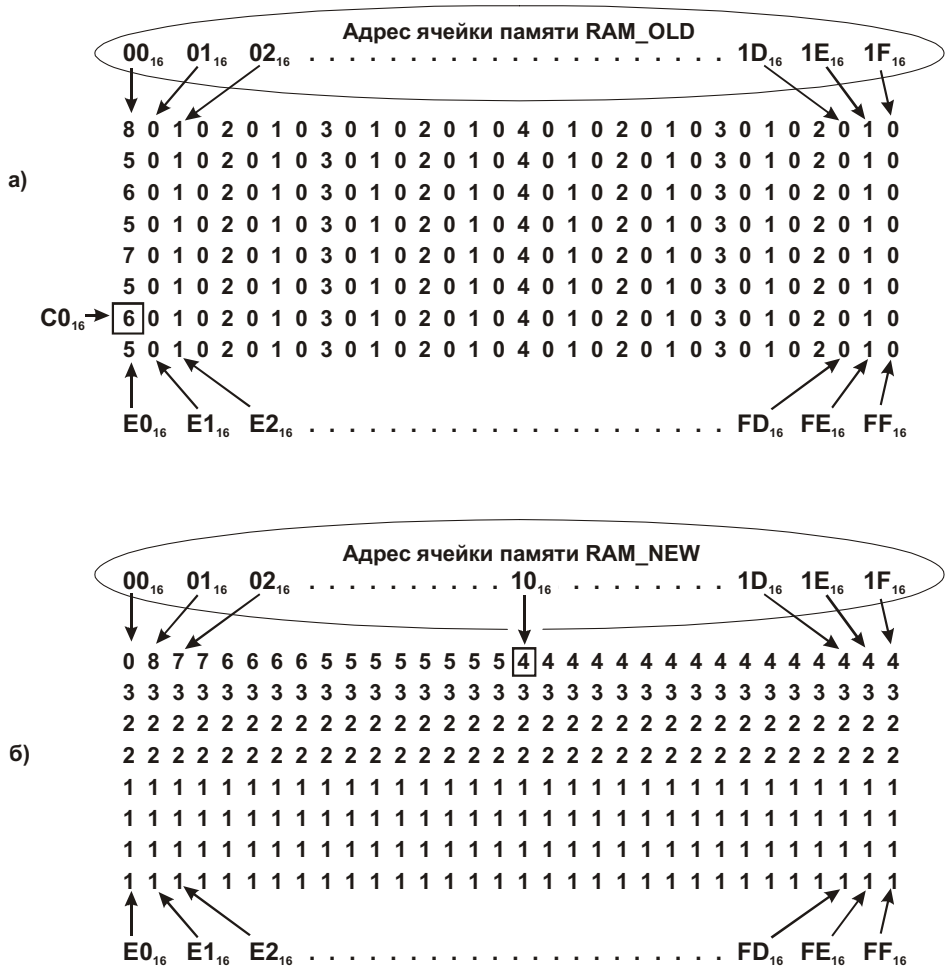

Рис. 7.10. Кодировка блоков памяти RAM_OLD (а) и RAM_NEW (б)

Отметим, что в данном примере левая граница флага “размыта”, т. е. может сливаться с цепочкой соседних нулевых битов, как в только что рассмотренном примере. Поэтому определение положения флага по его левой границе в данном случае невозможно.

В [55] рассмотрены также более сложные решения, при которых флаг размещается в нескольких байтах.

С точки зрения пользователя транспортной системы флаг представляется лишним элементом, и поэтому неплохо было бы если не избавиться от него (как ни странно, и это возможно, см. п. 7.4), то хотя бы уменьшить его длину, чтобы повысить скорость передачи данных.

7.2. Минимизация длины флага начала кадра

В рассмотренных далее решениях (п. 7.2.1 и 7.2.2) длина флага уменьшена до одного бита, при этом длина кадра постоянна. Разумеется, такой флаг неуникален. После установления синхронизации приемнику точно известно ожидаемое положение очередного флага и вычислено его значение (0 или 1). В первом решении значения флаговых битов чередуются во времени; во втором – представлены псевдослучайной последовательностью битов, что позволяет вести нумерацию кадров.

7.2.1. Использование чередующихся нулевых и единичных битов в качестве флагов начала кадров

Предлагаемое решение задачи минимизации длины флага основано на использовании чередующихся нулевых и единичных битов в качестве флагов начала кадров [14] и поясняется схемой, приведенной на рис. 7.11. Низкоскоростные потоки данных из каналов 1 – N поступают на входы мультиплексора MUX 1 (MUX 2). Сумма этих потоков в виде последовательности кадров пересылается с относительно высокой скоростью по линии связи к удаленному мультиплексору MUX 2 (MUX 1), который распознаёт кадр и распределяет данные по соответствующим каналам.

В предлагаемом решении битстаффинг не нужен, поэтому передаваемый по линии кадр имеет фиксированную длину, флаг представлен одним битом в каждом кадре.

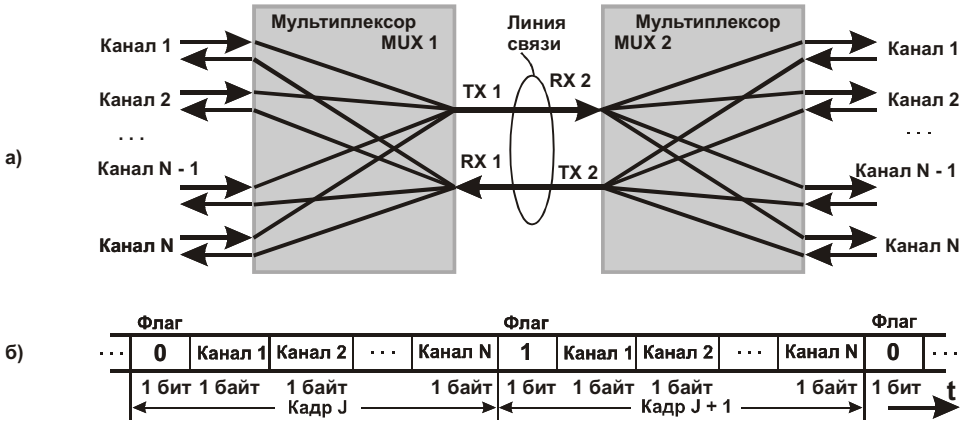

рис. 7.11. Мультиплексная передача данных с временным разделением каналов: *а* – структурная схема; *б* – структура одного из потоков данных в линии связи (TX 1 → RX 2 или TX 2 → RX 1)

Идея заключается в следующем. В установившемся режиме после вхождения мультиплексоров в логическое соединение, а затем и в кадровый синхронизм каждый мультиплексор помечает передаваемые им в линию последовательные кадры чередующимися одноразрядными флаговыми битами, как показано на рис. 7.11, *б*. Некоторый кадр с номером J (нумерация условная) содержит флаг, равный лог. 0. Следующий кадр с номером J + 1 имеет флаг, равный лог. 1. Последующие кадры содержат флаги, равные соответственно лог. 0, лог.1, лог. 0 и т. д.

Принимая кадр из линии, каждый мультиплексор проверяет соответствие полученного значения флагового бита ожидаемому. Например, после получения кадра с флаговым битом, равным лог. 0, предсказывается поступление кадра с флаговым битом, равным лог. 1. Позиция ожидаемого флагового бита известна, так как кадр имеет фиксированную длину, а предыдущая точка отсчета границы кадра уже отслежена в предыстории.

В отсутствие ошибок наблюдается правильное чередование флаговых битов. Ошибки, например проскальзывания, в конечном счете приведут к нарушению такого чередования и, следовательно, будут обнаружены, но, возможно, не сразу. При этом синхронизация будет временно потеряна и затем вновь восстановлена.

Вхождение мультиплексоров в кадровую синхронизацию

Для начала нужно установить связь между мультиплексорами по линии. Линия может быть выделенной или коммутируемой, содержащей ретрансляторы или иные устройства.

Каждый мультиплексор (MUX 1, MUX 2) имеет встроенный модем, подключенный к линии (модемы на рисунках не показаны). Один из модемов является вызывающим, т. е. инициатором установления связи, другой – вызываемым. В результате начального взаимодействия модемов (например по протоколу V.42 XID) и переговоров между ними, в некоторый момент мультиплексоры оказываются на связи и готовы передавать друг другу данные.

Нас сейчас интересует последующая фаза взаимодействия между мультиплексорами – установление кадровой синхронизации.

Последовательность событий такова.

1. Сразу после установления связи между модемами мультиплексоры начинают посылать друг другу непрерывные и неструктурированные потоки G, состоящие из лог. 1 (рис. 7.12, *а*, рис. 7.13).

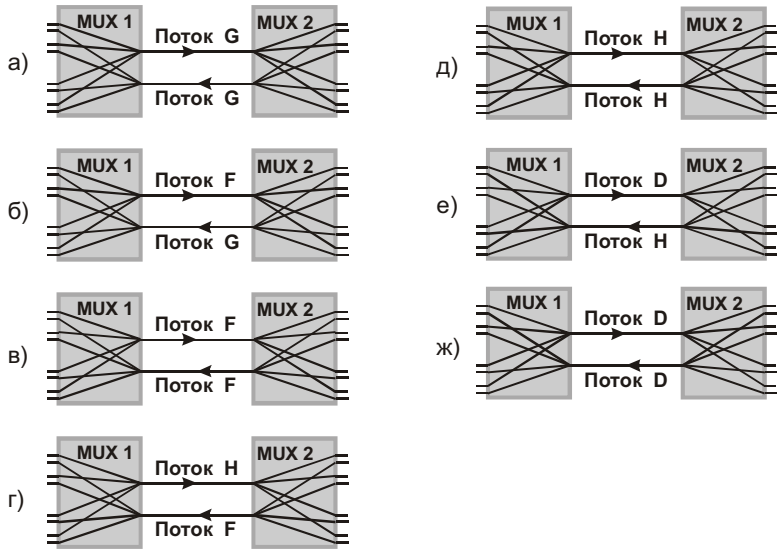

Рис. 7.12. Стадии *а – ж* установления кадровой синхронизации между мультиплексорами MUX 1 и MUX 2. Структура потоков F, G, H и D приведена на рис. 7.13

2. Один из мультиплексоров, например тот, по чьей инициативе организована связь (в данном примере мультиплексор MUX 1), через некоторое время начинает формировать поток F (см. рис. 7.12, *б*, рис. 7.13). Мультиплексор MUX 2 по-прежнему выдает поток G.

3. Мультиплексор MUX 2 обнаруживает в поступающем на его вход потоке F периодическое появление комбинаций 1011111101 и расценивает второй лог. 0 каждой такой комбинации как флаг начала смежной пары кадров. Зная длину кадра, мультиплексор MUX 2 расставляет недостающие (единичные) флаги, как это сделано на рис. 7.13. Задача наполовину решена – мультиплексор MUX 2 теперь способен распознавать границы кадров, присылаемых мультиплексором MUX 1. Факт установления кадровой синхронизации по направлению MUX 1 $\rightarrow$ MUX 2 внешне проявляется в том, что мультиплексор MUX 2 вместо потока G начинает выдавать поток F (см. рис. 7.12, *в*).

4. Мультиплексор MUX 1 обнаруживает в поступающем на его вход потоке F периодическое появление комбинаций 1011111101 и расценивает второй лог. 0 каждой такой комбинации как флаг начала смежной пары кадров. Зная длину кадра, мультиплексор MUX 1 расставляет недостающие (единичные) флаги, как это сделано на рис. 7.13. Теперь синхронизация установлена в обоих направлениях, и можно начинать переход к рабочему режиму. Для этого мультиплексор MUX 1 оповещает партнера об успешном установлении двусторонней кадровой синхронизации переходом к выдаче

потока H с сохранением достигнутой ранее кадровой синхронизации (рис. 7.13). Новое состояние системы показано на рис. 7.12, *г*.

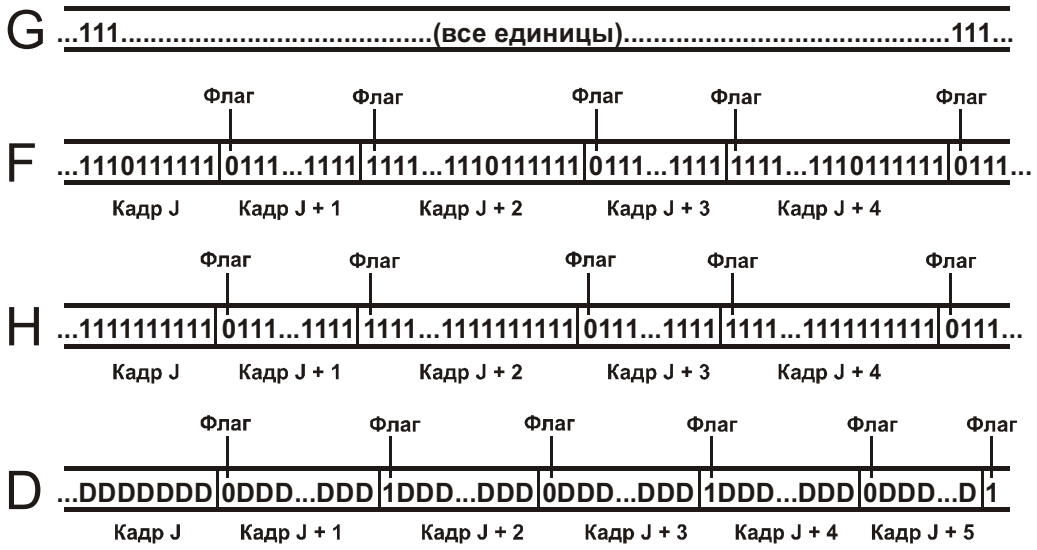

Рис. 7.13. Потоки фиксированных данных (диаграммы G, F, H), используемые на этапе установления синхронизации, и поток полезных данных (диаграмма D)

5. Мультиплексор MUX 2 подтверждает готовность перехода к рабочему режиму выдачей партнеру потока H с сохранением достигнутой ранее кадровой синхронизации (см. рис. 7.12, *д*).

6. Мультиплексор MUX 1 обнаруживает на входе поток H и переходит к выдаче потока D (см. рис. 7.13) с сохранением достигнутой ранее кадровой синхронизации. Этот поток соответствует рабочему режиму временного разделения каналов 1 – N (см. рис. 7.11). Состояние системы показано на рис. 7.12, *е*.

7. Мультиплексор MUX 2 через некоторое время после начала выдачи потока H (см. п. 5) переключается на выдачу потока D (см. рис. 7.13) с сохранением достигнутой ранее кадровой синхронизации. Этот поток соответствует рабочему режиму временного разделения каналов 1 – N (см. рис. 7.11). Состояние системы показано на рис. 7.12, *ж*.

Таким образом, синхронизация достигнута, система обслуживает каналы 1 – N.

Отметим, что каждый мультиплексор содержит внутренний таймер, который на соответствующих этапах установления синхронизации контролирует время ожидания правильных ответных реакций партнера. При превышении допустимого времени ожидания мультиплексор переходит к выдаче партнеру неструктурированного потока G (см. рис. 7.13). Партнер теряет синхронизацию и также переходит к выдаче потока G. В результате оба мультиплексора оказываются в рассмотренном ранее исходном состоянии (см. рис. 7.12, *а*), и попытка вхождения в синхронизм повторяется.

Потеря и восстановление синхронизации

Если в процессе или после установления взаимной синхронизации мультиплексоров хотя бы один из них обнаруживает неправильную последовательность флаговых битов (отличную от последовательности вида ...010101...), то это означает, что синхронизация потеряна.

В качестве примера поведения системы при потере синхронизации рассмотрим последовательность событий, приведенную на рис. 7.14.

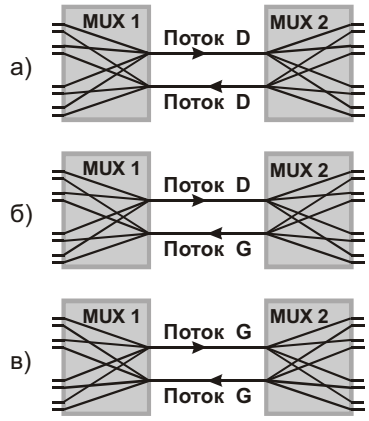

Рис. 7.14. Последовательность состояний системы при потере синхронизации мультиплексором MUX 2

1. В исходном состоянии (рис. 7.14, *а*) оба мультиплексора находятся в рабочем режиме и формируют потоки D (см. рис. 7.13 и рис. 7.11).

2. В некоторый момент мультиплексор MUX 2 обнаруживает ошибку в последовательности полученных флаговых битов: вместо ожидаемого лог. 0 принята лог. 1, или наоборот. Так как мультиплексор MUX 2 не может справиться с проблемой в одиночку (у него нет ориентиров для поиска очередного флагового бита), он обращается за помощью к мультиплексору MUX 1 выдачей ему неструктурированного потока G (рис. 7.14, *б*).

3. Мультиплексор MUX 1 не обнаруживает во входном потоке G флаговых битов и теряет синхронизацию с мультиплексором MUX 2 (столь грубая просьба о помощи воспринята). В ответ на потерю синхронизации мультиплексор MUX 1 посылает в мультиплексор MUX 2 поток G. Таким образом, ситуация усугубляется – синхронизация отсутствует уже в обоих направлениях (рис. 7.14, *в*). Мультиплексоры передают друг другу потоки G. А это и есть начальное условие для запуска уже рассмотренной процедуры установления синхронизации (см. рис. 7.12, *а*).

Через определенное время, заданное таймером мультиплексора MUX 1, система переходит в состояние, показанное на рис. 7.12, *б*, и т. д. Процесс завершается полным восстановлением синхронизации (см. рис. 7.12, *ж*).

Отметим, что для установления синхронизации можно воспользоваться методом статистического анализа потока принимаемых данных (см. п. 7.3).

7.2.2. Использование псевдослучайных битов в качестве флагов начала кадров

В только что описанной системе передачи данных последовательность передаваемых кадров сопровождается чередующимися одноразрядными флаговыми битами: “…010101…”. В установившемся режиме приемник следит за правильностью следования флаговых битов и обрабатывает принимаемые кадры. Однако однородный поток кадров в большинстве случаев необходимо структурировать, т. е. помечать в нем более крупные информационные объекты, например, группы кадров. Начало группы можно помечать, например, установкой в единицу некоторого служебного бита в заголовке кадра, но такое решение неэкономично. Нельзя ли совместить использование одноразрядных флаговых битов с разметкой относительного положения кадров в их непрерывном потоке?

Чтобы решить эту задачу, следует как-то преобразовать однородную последовательность “…010101…”. Для введения признака начала группы кадров можно было бы

ввести в эту последовательность преднамеренное нарушение, например заменой нуля единицей или наоборот. Однако такая замена не обеспечивает хорошей отличимости внесенного нарушения от искажений сигнала в результате действия помех в линии. Еще одна проблема заключается в том, что аппаратура формирования кадровых битов и их распознавания должна быть достаточно простой, процесс вхождения в синхронизацию должен обладать быстрой сходимостью. Этим требованиям отвечает рассмотренное далее решение [75, 76], рис. 7.15.

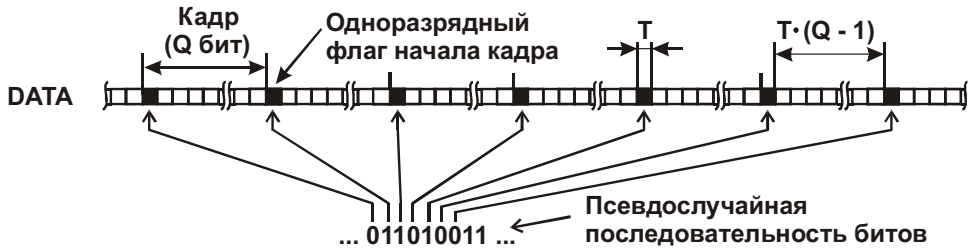

Рис. 7.15. Схема, поясняющая идею использования псевдослучайных битов в качестве флагов начала кадров; DATA – поток битов между передатчиком и приемником

Так же, как и в предыдущем решении (см. рис. 7.11) начало кадра обозначается флаговым битом. Но в данном случае цепь из флаговых битов нетривиальна по структуре и формируется с помощью генератора псевдослучайной последовательности битов (рис. 7.16).

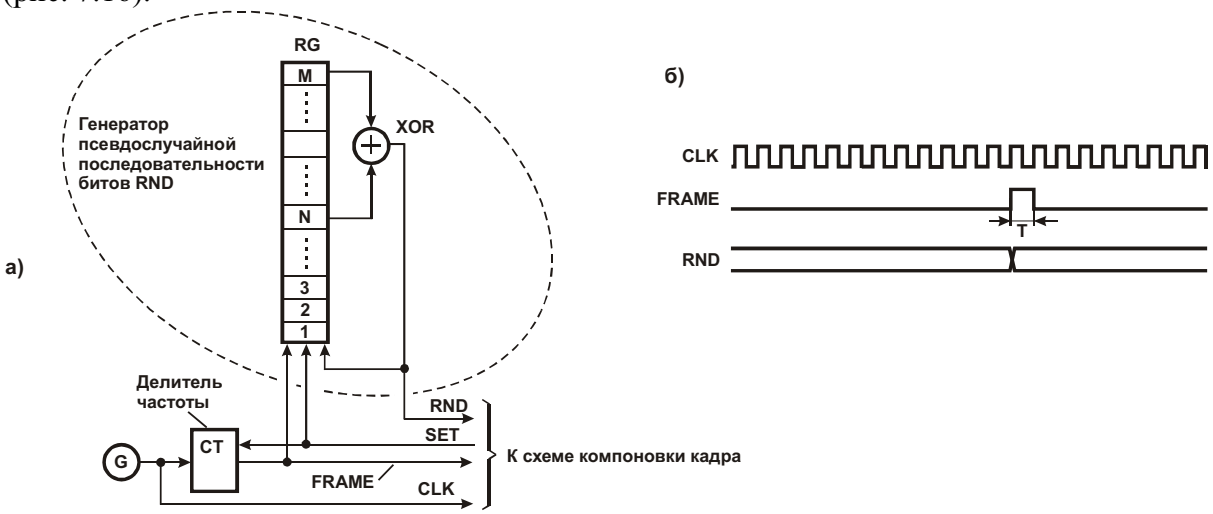

Рис. 7.16. Формирование флаговых битов: *а* – схема; *б* – временная диаграмма

Предположим, что число битов в кадре равно Q, частота сигнала CLK от генератора G, задающего скорость передачи данных, равна F Гц. Длительность T битового интервала составляет 1/F с. Генератор псевдослучайной последовательности битов RND построен по классической схеме (она подробно рассмотрена в п. 8.4.1) на основе M-разрядного сдвигового регистра RG и элемента Исключающее ИЛИ (XOR), входы которого соединены с выходами разрядов M и N регистра. Выход элемента XOR соединен с входом данных регистра RG. Делитель частоты (счетчик по модулю Q) снижает частоту F сигнала CLK в Q раз и формирует кадровый импульс FRAME длительностью T, который поступает в схему компоновки кадра и одновременно воздействует на вход синхронизации регистра RG генератора псевдослучайной последовательности битов. По положительному фронту сигнала FRAME этот генератор формирует очередной псевдослучайный бит RND. Схема компоновки кадра при обнаружении импульса FRAME считывает бит RND и использует его для вставки очередного флага в поток данных, как было показано на рис. 7.15.

По сигналу SET регистр RG устанавливается в состояние 111…1, а делитель частоты – в нулевое состояние. Такая установка позволяет задать исходную точку в системе псевдослучайного отсчета передаваемых кадров. В этой системе отсчета последовательность кадров неявно нумеруется кодами в регистре RG. Эти коды формируются в заранее известном и строго определенном порядке, зависящем от разрядности регистра и точки подключения к нему обратной связи. Так, предварительно установленный в состояние 111…1 регистр RG после воздействия первого импульса FRAME переходит в состояние 011…1 (первый разряд – слева), после второго импульса – в состояние 001…1 и т. д. В полном цикле работы генератора псевдослучайной последовательности битов в регистре однократно формируются все возможные коды, за исключением нулевого. Полученные таким способом M-разрядные номера кадров в явном виде не передаются в удаленный приемник, но могут быть восстановлены им при реставрации псевдослучайной последовательности битов, как будет показано далее.

В протоколе обмена данными номера кадров определяют их статус. Например, при M = 5 по линии связи непрерывно передаются группы из $2^5 - 1 = 31$ кадра. В каждой группе кадр с номером 11111_2 рассматривается как начальный. Приемная аппаратура распознаёт эти кадры по их номерам и трактует их содержимое по-разному, в соответствии с принятым протоколом.

Сформированная последовательность кадров с одноразрядными флаговыми битами (рис. 7.15) передается по линии связи в приемную аппаратуру. Для распознавания границ и номеров кадров применяется схема (рис. 7.17) на основе такого же генератора псевдослучайной последовательности, как и в схеме формирования флаговых битов.

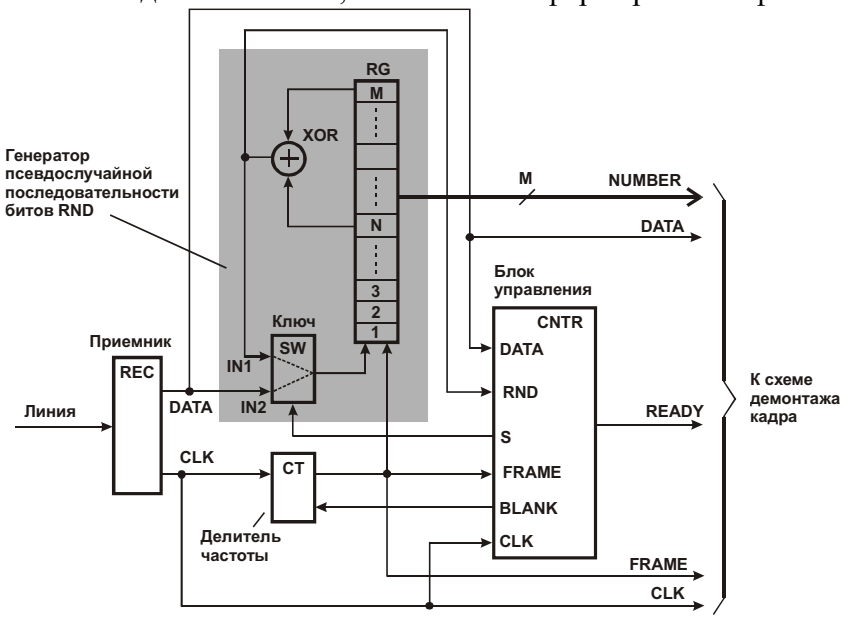

Рис. 7.17. Схема распознавания межкадровых границ

Приемник REC усиливает принятый сигнал и выделяет из него синхросигнал CLK и биты DATA – собственно данные и вставленные в нужных местах псевдослучайные биты флагов начала кадров (см. рис. 7.15). Делитель частоты выполнен в виде счетчика по модулю Q (напомним, что Q – число битов в кадре). После установления правильной синхронизации восстановленные сигналы CLK, FRAME (признак наличия в потоке флагового бита) и RND (восстановленный псевдослучайный бит) находятся в тех же временных соотношениях, что и соответствующие сигналы в схеме формирования флаговых битов (см. временные диаграммы на рис. 7.16).

Сигнал BLANK длительностью в один период T сигнала CLK в необходимых случаях формируется блоком управления CNTR и притормаживает работу делителя

частоты на один такт. Это позволяет смещать импульс FRAME на один такт вправо относительно "неподвижной" последовательности битов DATA при поиске истинного положения флагового бита. Блок CNTR формирует также сигнал S управления электронным ключом SW. На этапе установления синхронизации ключ транслирует на выход сигнал с входа IN2. Когда синхронизация установлена, ключ передает на выход сигнал с входа IN1, т. е. переводит генератор псевдослучайной последовательности битов в автономный режим, при котором он нечувствителен к сигналам DATA. Параллельный M-разрядный код NUMBER на выходах регистра RG при правильной синхронизации отображает псевдослучайный номер кадра и с некоторой задержкой повторяет код в аналогичном регистре схемы формирования флаговых битов (см. рис. 7.16). Пока синхронизация не установлена, сигнал READY = 0; после установления синхронизации READY = 1.

Процесс установления синхронизации сопровождается серией довольно длительных экспериментов, которые проводятся приемной аппаратурой (эта аппаратура, разумеется, должна обладать достаточным "интеллектом"). Перед началом каждого эксперимента делается предположение о том, что некоторая периодически повторяющаяся позиция в потоке битов соответствует флаговой. Конечно, вероятность попадания в правильную сетку размещения флаговых битов с одной попытки мала и составляет 1/Q. Но если "повезет", то первый же эксперимент позволит войти в синхронизацию, т. е. правильно ориентироваться в потоке кадров, отслеживая их границы и номера. Если первый эксперимент оказался неудачным, то предполагаемая сетка размещения флаговых битов смещается блоком управления на один бит вправо. Проводится второй эксперимент и т. д. В худшем случае правильная синхронизация окажется возможной только после проведения последнего эксперимента с номером Q.

Предположим, что в начале эксперимента синхронизации нет, сигнал READY = 0, регистр RG и делитель частоты находятся в произвольных состояниях. Блок управления CNTR переводит ключ SW в режим передачи сигналов с входа IN2. При прохождении определенной позиции каждого кадра (эта позиция "подозревается" в качестве носителя флагового бита) на выходе делителя частоты формируется импульс FRAME. По его положительному фронту в первый разряд регистра RG принимается бит – кандидат на звание флагового. Одновременно с этим остальные биты этого регистра сдвигаются вверх на один разряд, бит из разряда M теряется.

По прошествии M кадров регистр RG заполняется принятыми битами, "подозреваемыми" в выполнении роли флаговых. Теперь возможны две ситуации.

1. Регистр действительно заполнен флаговыми битами группы из M соседних кадров. В этой ситуации содержимое регистра RG с некоторой задержкой повторяет код в аналогичном регистре схемы формирования флаговых битов (см. рис. 7.16). Существенно, что всякий раз в момент записи очередного бита в первый разряд регистра RG сигналы RND и DATA на входах IN1 и IN2 ключа SW совпадают. Это происходит благодаря тому, что в передающей и приемной аппаратуре применены одинаковые генераторы псевдослучайных последовательностей битов. Поэтому сигнал на входе IN1 формируется по тем же правилам, которым следует сигнал на входе IN2. Иными словами, сигнал RND можно рассматривать как результат предсказания значения очередного флагового бита, поступившего из линии.

Блок управления следит за совпадением предсказанных и фактически принятых флаговых битов. Если предсказания постоянно оправдываются при прохождении, например, 100 кадров, то предполагается, что синхронизация достигнута, сигнал READY устанавливается в состояние лог. 1. Блок управления переводит ключ SW в режим передачи сигнала с верхнего входа (IN1), так что генератор псевдослучайной последовательности битов переходит к автономной работе и не реагирует на ошибки, которые могут появляться в сигнале DATA. Это создает некоторую разумную инерционность

системы синхронизации и предотвращает выход и повторное вхождение в синхронизм при допустимом уровне ошибок в линии.

Теперь для дальнейшей обработки кадра имеется вся необходимая информация. Обработка, прежде всего, заключается в демонтаже кадра: вычеркивании флагового бита, определении номера кадра в группе, проверке и вычеркивании контрольных сумм и т. д.

При правильной синхронизации блок управления CNTR продолжает сопоставлять предсказанные и фактически принятые флаговые биты. Интенсивность несовпадений битов в этой ситуации не должна превышать некоторую заданную величину. Например, допустимым может считаться одно несовпадение на 100 тыс. кадров, в противном случае качество системы передачи данных расценивается как низкое. При резком повышении интенсивности несовпадений битов синхронизация считается потерянной, сигнал READY переводится в состояние лог. 0 и блок управления начинает новую серию экспериментов по восстановлению синхронизации.

2. Регистр заполнен не флаговыми битами. Эта ситуация приводит к множественным несовпадениям предсказанных и фактически полученных битов, причем вероятность формирования непрерывной последовательности ложных случайных совпадений быстро уменьшается со временем и с ней можно не считаться. Сигнал READY остается в нулевом состоянии, эксперимент признается неудачным. Блок управления CNTR формирует сигнал BLANK, который приостанавливает работу делителя частоты на один такт. После этого начинается новый эксперимент, при котором флаговые биты ищутся в соседних (сдвинутых вправо на один такт) позициях кадров и т. д.

Подводя итоги материалам, рассмотренным в п. 7.2, отметим следующее.

1. Длину флага можно сократить до одного бита, но для его распознавания нужен достаточно высокий “интеллект” приемной аппаратуры.

2. Использование псевдослучайных последовательностей флаговых битов позволяет не только распознавать границы кадров, но и вести их относительную нумерацию в пределах выделенных групп. Нумерация позволяет приемнику классифицировать принимаемые кадры по их номерам в соответствии с принятым протоколом обмена. Например, приемник может отличать “обычные” кадры от служебных, а среди них различать кадры различных типов. В многоканальной системе передачи данных (с мультиплексированием каналов) номер кадра позволяет приемнику определить, к какому каналу относятся содержащиеся в кадре данные, и т. д. Пример системы передачи данных, использующей псевдослучайные флаговые биты, рассмотрен в п. 7.9.

7.3. Использование раздробленного флага начала кадра

Вновь вернемся к вопросу синхронизации приемника с передатчиком. Напомним, что поток передаваемых по линии связи битов состоит из структурных единиц (например кадров). Идея построения “обычного” кадра поясняется рис. 7.18. Кадр представляет собой группу битов с границами p и q. Флаг размещен в начале кадра, представлен уникальным кодом и имеет фиксированную длину; его границы – a и b. За флагом следуют слова b – c, c – d, d – e и т. д.

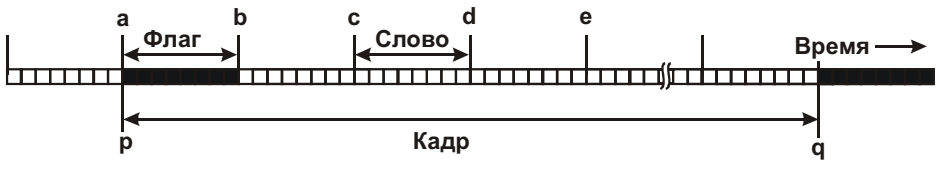

рис. 7.18. Пример структуры кадра.

Как следует из примера, рассмотренного в п. 7.2, длину флага можно сократить до одного бита, при этом битстаффинг не нужен. Здесь мы приведем родственное, но более общее решение, в котором многоразрядный флаг существует, но в виде разобщенных битов. Такое решение позволяет быстро устанавливать синхронизацию между устройствами и сохранять ориентиры (биты флага) при частичном повреждении кадра из-за помех в линии.

7.3.1. Применение неуникального флагового кода

Рассмотрим предлагаемую структуру кадра (рис. 7.19) [20].

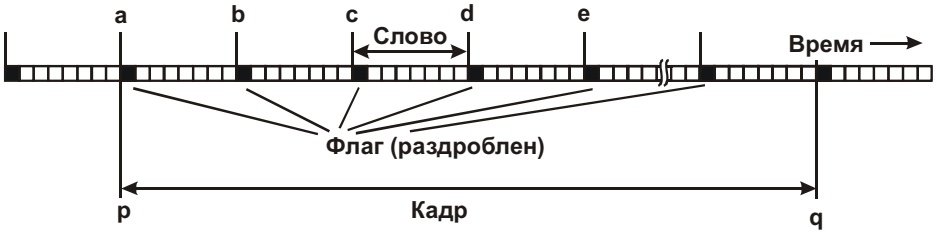

рис. 7.19. Структура кадра с раздробленным на отдельные биты флагом

В отличие от традиционной структуры кадра (рис. 7.18), в данном случае флаг фиксированной длины раздроблен на отдельные биты. Эти биты дополнительно служат метками начала слов кадра. В то же время совокупность флаговых битов позволяет достаточно надежно распознать кадр как целое. Длина слова фиксирована, битстаффинг не нужен.

Приемник распознаёт флаг с использованием вероятностных оценок. При анализе входного потока битов он выявляет в нем устойчивые закономерности. Проще говоря, приемник как бы просматривает поток, показанный на рис. 7.19, сквозь непрозрачную маску, в которой вырезаны отверстия, соответствующие черным квадратикам на рисунке. Маска может накрывать, например, 10 кадров, так что при восьмиразрядном флаге в прорези маски попадает цепочка из $10 \times 8 = 80$ битов.

Шаг за шагом перемещая маску вдоль исследуемого потока, не позже чем через семь шагов увидим сквозь ее отверстия правильную флаговую комбинацию битов, повторенную 10 раз, что с высокой вероятностью подтверждает факт обнаружения исходной разметки кадров. (Как видим, число шагов поиска невелико, что способствует быстрому установлению синхронизации приемника с передатчиком.) В отсутствие ошибок передачи вероятность ложного обнаружения искомой цепочки из 80 битов в случайном потоке данных составляет 2^{-80} и уменьшается по мере дальнейшего перемещения маски. Это справедливо при использовании скремблера данных со стороны передатчика (см. п. 8.4). (Напомним, что скремблер передатчика преобразует данные таким образом, что они выглядят как случайные битовые последовательности с равновероятным появлением лог. 0 и лог 1. Дескремблер приемника выполняет обратное преобразование.)

7.3.2. Построение кросс-корреляционной матрицы для распознавания раздробленного флага

Рассмотрим процесс распознавания границ кадров с демонстрацией последовательности поиска на простом примере.

Предположим, что передатчик формирует непрерывную последовательность кадров со структурой, показанной в верхней части рис. 7.20.

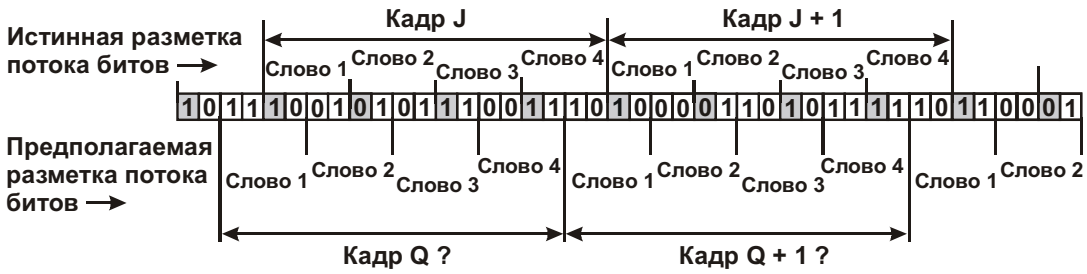

Рис. 7.20. Упрощенный пример разметки потока битов на кадры и слова. Флаговые биты (рассредоточенные коды 1011_2) выделены серым фоном

Задача приемника состоит в том, чтобы путем набора и анализа статистической информации о входном потоке данных выявить границы кадров. При этом заранее известно, что: 1) кадр состоит из четырех 4-разрядных слов; 2) флаг представлен кодом 1011_2, причем первым передается крайний левый бит.

Процесс поиска начинается с того, что приемник выдвигает предположение о том, что разметка потока битов уже известна и соответствует приведенной в нижней части рис. 7.20. Эта разметка начинается из произвольно выбранной точки. Как видно из рисунка, в данном примере имеется сдвиг на два бита влево относительно истинного положения границ кадров. Тем не менее приемник пока оперирует кадрами Q, Q + 1, Q + 2 и т. д. Знаки вопросов на рисунке отражают тот факт, что синхронизация еще не достигнута.

Кадр содержит 16 бит. Для удобства анализа он представлен матрицей размерностью 4 × 4, как показано в верхней части рис. 7.21.

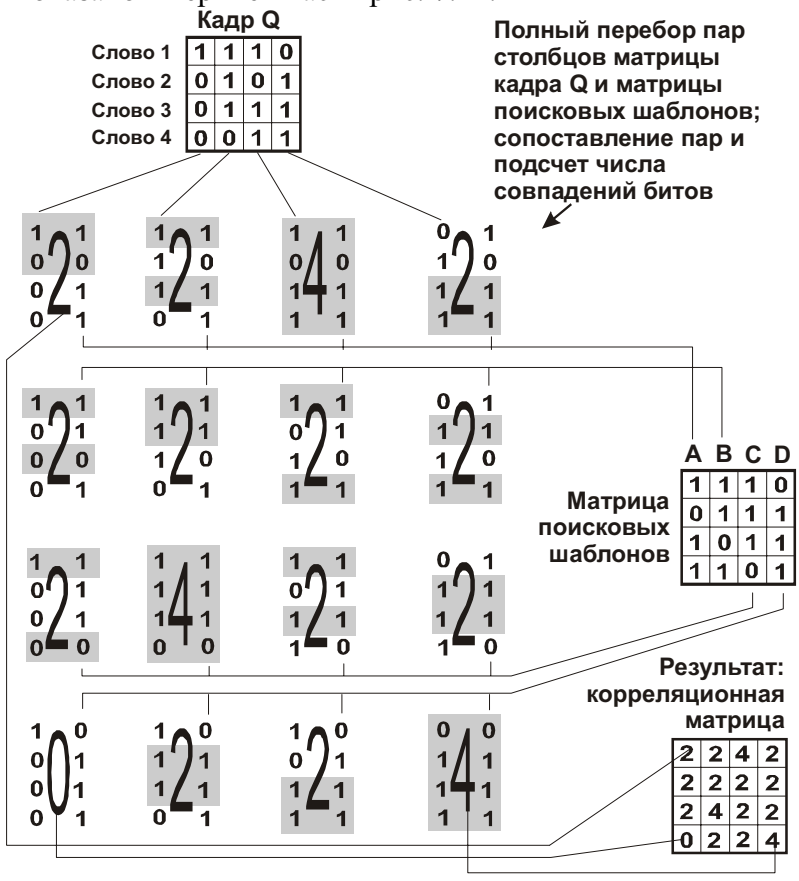

Рис. 7.21. Схема вычисления корреляционной матрицы кадра Q. Серым фоном и укрупненными цифрами выделены совпадения и их число

Проведем мысленный эксперимент. Предположим, что кадр Q занял правильное положение, совпадающее с положением кадра J. Тогда искомая флаговая комбинация битов 1011 в матрице кадра Q разместится на своем месте – в первом (крайнем левом) столбце. При смещении кадра Q относительно кадра J влево на одну битовую позицию флаговая комбинация битов 1011 сместится во второй столбец матрицы. Дальнейшее смещение на одну позицию приводит к ситуации, отображенной на рис. 7.20 и рис. 7.21, когда искомый столбец матрицы – третий. Еще одно смещение – и флаг перемещается в четвертый столбец.

При следующем перемещении кадра Q влево на одну битовую позицию флаг вновь оказывается в первом столбце матрицы, но он циклически сдвинут на один разряд вправо и теперь имеет вид 1101. Продолжая серию перемещений кадра можно убедиться, что флаг может также приобретать формы 1110 и 0111. Таким образом, с учетом неопределенности выбора точки отсчета, флаговые (статистически устойчивые) комбинации битов в "системе координат" приемника могут быть такими: 1011, 1101, 1110 или 0111. Эти равноправные комбинации битов для опознания флага представлены на рис. 7.21 матрицей поисковых шаблонов размерностью 4×4. В этой матрице столбец A соответствует истинному флагу, а остальные столбцы (B, C, D) – его циклически сдвинутым копиям.

Итак, имеются две матрицы: первая отображает содержимое кадра Q, вторая представляет собой набор из четырех шаблонов для поиска.

Задача состоит в том чтобы оценить, какой шаблон (A, B, C или D) наиболее подходит кадру. Получив самый подходящий шаблон и зная к какому именно столбцу он тяготеет, можно после аналогичного анализа ряда кадров откорректировать принятую разметку потока битов, т. е. выровнять кадры Q с кадрами J.

Задача "примерки" шаблонов решается методом простого перебора вариантов с регистрацией числа благоприятных исходов (совпадений кодов) в корреляционной матрице.

Начнем перебор. Выберем из матрицы поисковых шаблонов столбец A и поочередно сопоставим с ним все столбцы матрицы кадра Q. Первый (левый) столбец матрицы кадра Q (код 1000) частично совпадает со столбцом A (код 1011), число совпадений битов равно двум. Зафиксируем этот результат в верхней строке и первом (левом) столбце корреляционной матрицы.

Второй столбец матрицы кадра Q (код 1110) также частично совпадает со столбцом A (код 1011), число совпадений битов тоже равно двум. Зафиксируем этот результат в верхней строке и втором столбце корреляционной матрицы.

Продолжая этот процесс, заполним все позиции корреляционной матрицы. Можно ли сделать какие-то выводы? Пока лишь частичные, так как объем обработанной информации недостаточен для окончательной оценки ситуации.

Действительно, корреляционная матрица не содержит ярко выраженного элемента с максимальным численным значением. Имеем размытый максимум, представленный тремя элементами с численным значением, равным 4. Три шаблона из четырех оказались перспективными. Первый столбец матрицы кадра Q представляется безнадежно непохожим ни на один из шаблонов.

Чтобы получить больше информации, проанализируем подобным образом кадры Q + 1 и Q + 2 (рис. 7.22).

Как видим, корреляционная матрица кадра Q + 2 также имеет размытый максимум. Просуммируем соответствующие элементы трех корреляционных матриц и поместим результаты в соответствующие позиции кросс-корреляционной матрицы, отражающей обобщенный взгляд на три кадра.

Здесь вырисовывается уже более реалистичная картина опознания. Появилось большее число градаций переменных, сформировался максимум, равный 12 (отмечен

серым фоном). По трем кадрам, конечно, еще трудно судить о надежности результата, но напомним, что это всего лишь упрощенный пример.

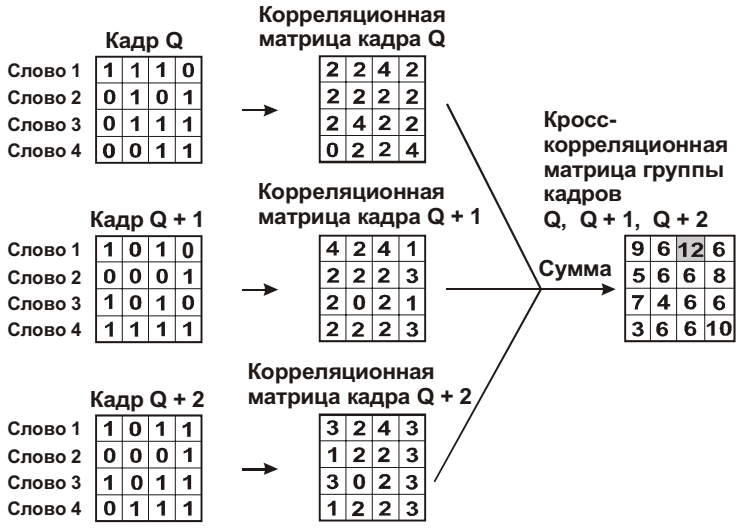

рис. 7.22. Схема вычисления кросс-корреляционной матрицы группы кадров Q, Q + 1, Q + 2

Максимальное число (12) находится на пересечении первой строки и третьего столбца. Возвращаясь к схеме формирования корреляционной матрицы (см. рис. 7.21) можно утверждать, что наиболее подходящий эталон равен 1011, а наиболее подозреваемым в переносе флага является третий столбец матрицы Q. Одновременно этим двум условиям удовлетворяет единственный вариант смещения кадра Q относительно кадра J, а именно, показанный на рис. 7.20. Остается сдвинуть кадр Q (и все последующие) на две битовые позиции вправо, после чего выявляются истинные границы кадров.

Подведем некоторые итоги. В рассмотренных структурах потоков данных (п. 7.1 – 7.3) флаг присутствует в явном виде и задается одним битом или раздробленной группой битов. Однобитовый флаг позволяет распознавать межкадровые границы и даже нумеровать кадры. Флаг, представленный раздробленной группой битов, позволяет уменьшить время вхождения приемника в синхронизацию с передатчиком. Кроме того, отдельные биты раздробленного флага служат признаками начала слов данных кадра. Это позволяет приемнику более уверенно ориентироваться в потоке данных в условиях повышенного уровня помех в канале связи и локализовать вероятные ошибки на уровне слов, которые примыкают к искаженным флаговым битам. Это может оказаться полезным в некоторых применениях (например, при передаче речи) когда частично поврежденный кадр можно использовать, отбросив его испорченный фрагмент, вместо того чтобы отбрасывать полный кадр.

7.4. Распознавание ячеек ATM в битовом и байтовом потоках данных

7.4.1. Структура ячейки ATM

Технология ATM (Asynchronous Transfer Mode – асинхронный режим передачи) предусматривает размещение данных внутри 53-байтовых ячеек (рис. 7.23). Как показано на рисунке, ячейки следуют по линии непрерывным потоком, хотя возможны и иные варианты их транспортирования (об этом – позже).