Вероятностная синхронизация в телекоммуникационных системах: разграничение байтов в битовом потоке данных

Для согласования работы удалённых друг от друга устройств в предыдущих статьях [1, 2] предложено использовать "вероятностную" ("стохастическую") синхронизацию. По сути, разработан некий, на первый взгляд парадоксальный, способ одновременного получения удалёнными друг от друга устройствами меток времени "ниоткуда", в нечётко предсказуемые моменты. Имеется в виду, что эти метки не созданы "по воле человека", как создают, например, распознаваемые на удалённой стороне линии связи заголовки информационных кадров – заранее заданные последовательности служебных битов. В данном случае метки возникают "сами собой" и извлекаются из псевдошумового сигнала, полученного в результате скремблирования потока "полезных" данных. Скремблированный поток подчиняется законам теории вероятностей, поэтому в нём обязательно присутствуют любые заранее заданные коды. Упомянутые метки времени формируются в моменты одновременного (с учётом задержки передачи) обнаружения этих кодов удалёнными друг от друга устройствами. В данной статье вероятностная синхронизация применена для указания границ между байтами в битовом потоке данных.

С повышением скорости передачи данных по линиям связи возрастают требования к быстродействию передающей и приёмной аппаратуры. Чтобы снизить тактовую частоту при обработке принимаемого сигнала, необходимо преобразовать входящий битовый поток в байтовый. Для такого преобразования приёмник должен знать, где размещены границы между байтами в непрерывном потоке принимаемых битов. Указание границ между байтами – не совсем простая задача, требующая введения избыточности в битовый поток.

Так, согласно [3], к каждому байту в битовом потоке добавляется бит разграничения, полученный от генератора псевдослучайной последовательности битов. Приёмник данных обнаруживает биты разграничения благодаря их устойчивому совпадению с эталонной псевдослучайной последовательностью битов. Другой способ [4] разграничения байтов в битовом потоке также предусматривает добавление избыточного бита к каждому байту. Этот бит формируется дублированием и инвертированием нулевого бита передаваемого байта. В результате начало байта сопровождается передачей комбинаций битов 01 или 10. Приёмник данных обнаруживает биты разграничения и нулевые биты данных благодаря их статистически устойчивому совпадению с кодами 01 или 10. Оба способа введения битов разграничения сложны и неэкономичны – на каждые восемь битов данных приходится вводить один служебный разграничительный бит.

Предлагаемое далее решение уменьшает вносимую в битовый поток избыточность практически до нулевого уровня. Рассмотрим сначала общую идею применения вероятностной синхронизации для разграничения байтов в битовом потоке данных (рис. 1).

Передающая аппаратура содержит преобразователь P-S параллельного кода в последовательный и скремблер [1]. Приёмная аппаратура включает в себя дескремблер и преобразователь S-P последовательного кода в параллельный. Байты данных поступают на вход TxD системы передачи данных под управлением синхросигнала на выходе TxC и передаются на выход RxD под управлением синхросигнала RxC. Преобразователь P-S формирует непрерывный поток битов DATA, сопровождаемый синхросигналом CLK. Этот поток скремблируется и пересылается по линии связи в аппаратуру приёмной стороны. После дескремблирования битовый поток данных DATA* в сопровождении синхросигнала CLK* поступает в преобразователь S-P и передаётся на выход в виде последовательности байтов. Основная задача, как отмечалось, состоит в определении приёмной аппаратурой местоположения границ между байтами в битовом потоке. Для её решения используется вероятностная синхронизация.

Предположим, что в некоторый момент произошло одно из трёх несовместных случайных событий $S_1 – S_3$. Эти события, говоря более точно, порождаются кодовыми ситуациями в линии связи (стрелка A), но для общего описания идеи это несущественно. На

рис. 1 событие S_2 условно показано в виде вспышки света в точке, расположенной ближе к передающей аппаратуре, чем к приёмной. Вспышка регистрируется сначала передающей, а затем и приёмной аппаратурой. В результате регистрации этого события в скремблере и дескремблере формируются импульсы J и J*. Эти импульсы служат опорными метками, ориентирами во времени для указания положения границ между байтами в битовом потоке данных.

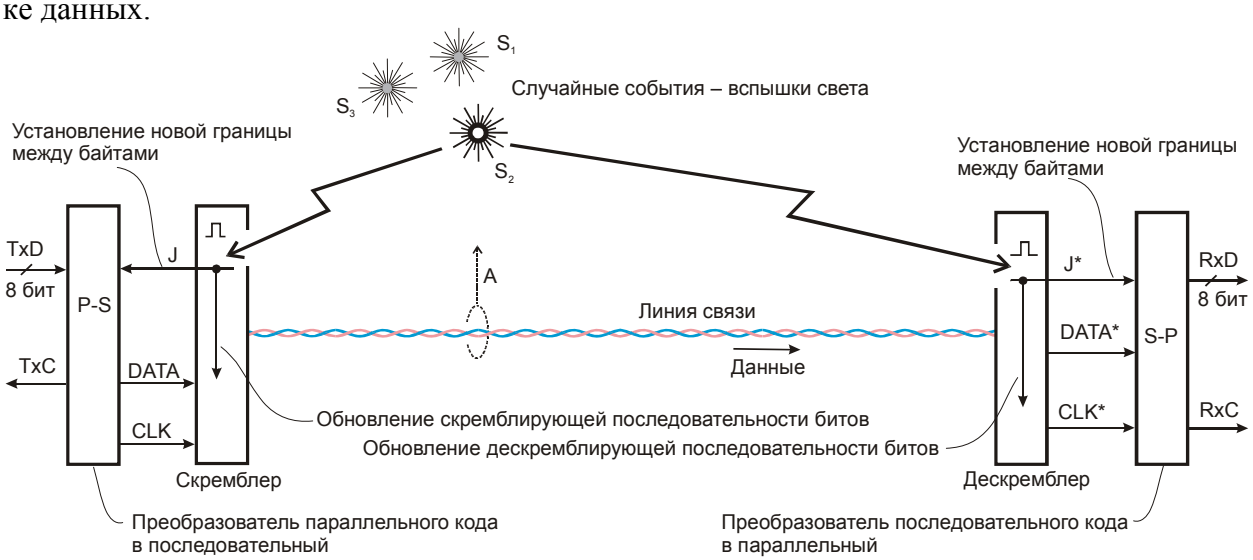


Рис. 1. Идея построения предлагаемой системы передачи данных

При правильной работе системы передачи данных и отсутствии импульсов J (J*) байты передаются с её входа на выход с равномерным темпом, как показано на рис. 2. То же происходит и в том случае, когда импульс J (J*) сформирован точно на границе между байтами. Тогда передающая аппаратура игнорирует его и продолжает непрерывное преобразование байтового потока данных в скремблированный битовый поток. Приёмная аппаратура также не реагирует на соответствующий импульс J*, если ранее была достигнута правильная синхронизация. В противном случае приёмная аппаратура синхронизируется импульсом J* и в дальнейшем правильно распознаёт положение границ между байтами.

Если импульс J (J*) поступает во время последовательной передачи по линии связи некоторого байта i (рис. 3), то передача прерывается и затем возобновляется в новой системе отсчёта времени, в которой нулевая точка соответствует моменту поступления этого импульса.

Помимо обозначения новых границ между байтами, импульсы J и J* синхронизируют работу скремблера и дескремблера. Каждое из событий S_i вызывает переход к соответствующим начальным точкам скремблирующей и дескремблирующей последовательностей битов (подробное описание этого процесса приведено в [1]). Это позволяет первоначально установить кодовую синхронизацию, а в случае её потери – автоматически восстановить правильную работу подсистемы “скремблер – дескремблер” при регистрации первого же события S_i.

Рассмотрим предлагаемую систему (рис. 4) подробнее. Начнём с описания её составных частей.

Сдвиговые регистры RG1 и RG2 предназначены для временного хранения фрагментов SDATA и SDATA* потока скремблированных данных и реализуют функцию “скользящего окна”, через которое просматривается битовая последовательность. В установившемся режиме эти фрагменты одинаковы (совпадают с точностью до задержки передачи). Приём очередного бита в регистр RG1 (RG2) происходит по положительному фронту сигнала на синхронизирующем входе C этого регистра. Одновременно с приёмом очередного бита с входа D ранее хранимые данные сдвигаются на один разряд вправо. В данном примере построения системы разрядность регистров RG1 и RG2 выбрана равной восьми, хотя она может быть большей или меньшей.

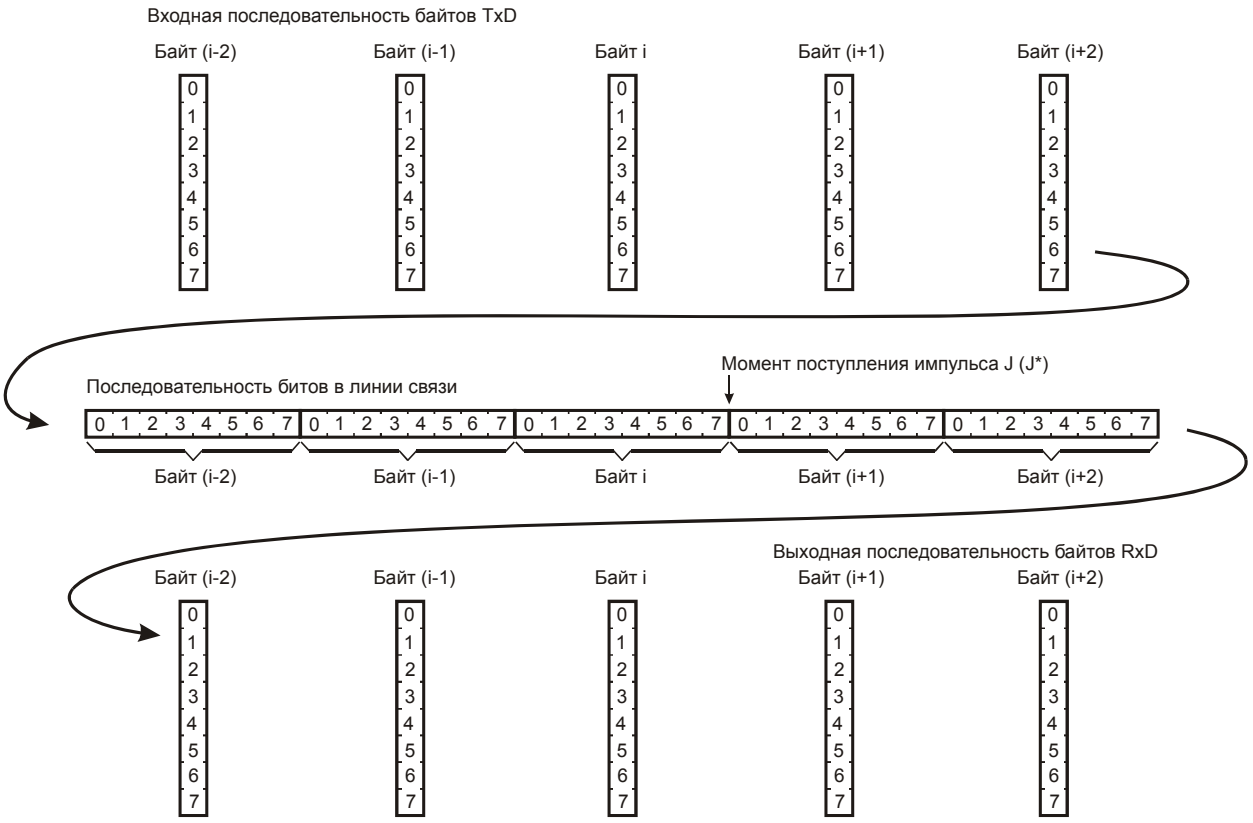


Рис. 2. Прохождение потока байтов через предварительно синхронизированную систему передачи в отсутствие импульса J (J*) или при его попадании на границу между байтами

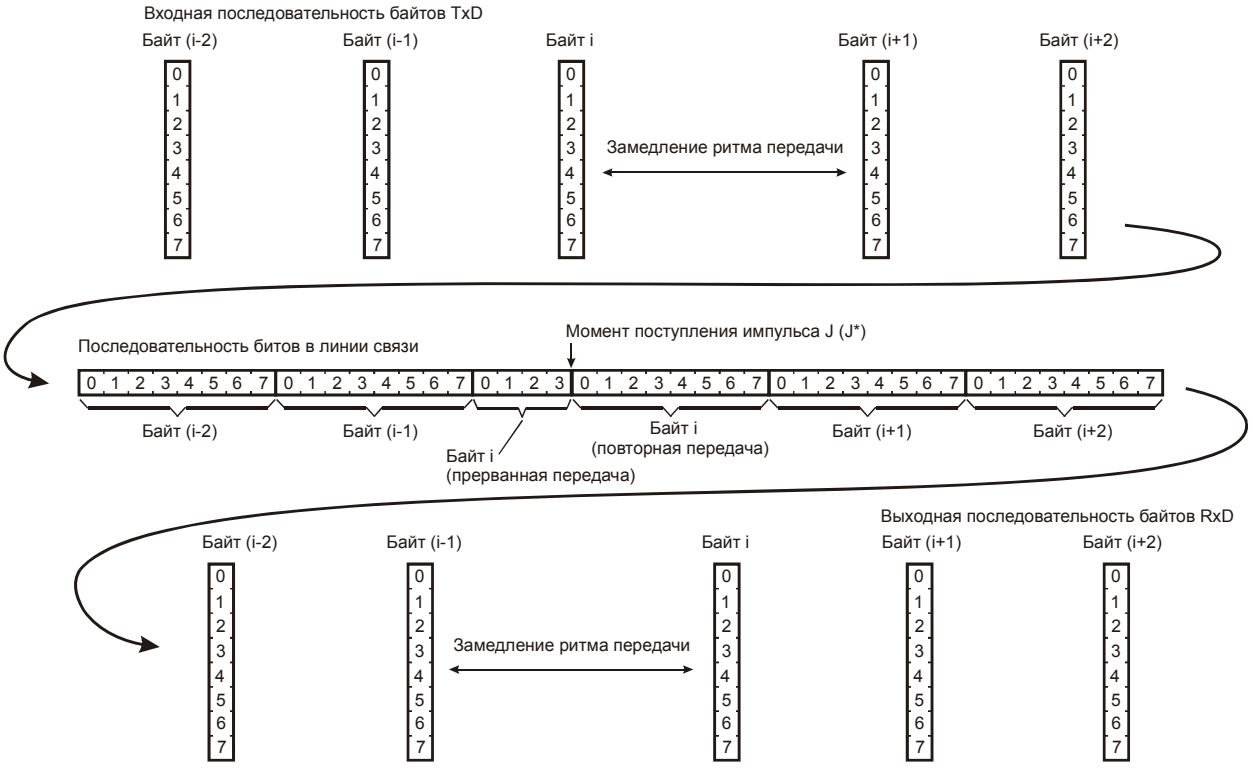


Рис. 3. Прохождение потока байтов через предварительно синхронизированную систему при поступлении импульса J (J*) во время передачи байта i

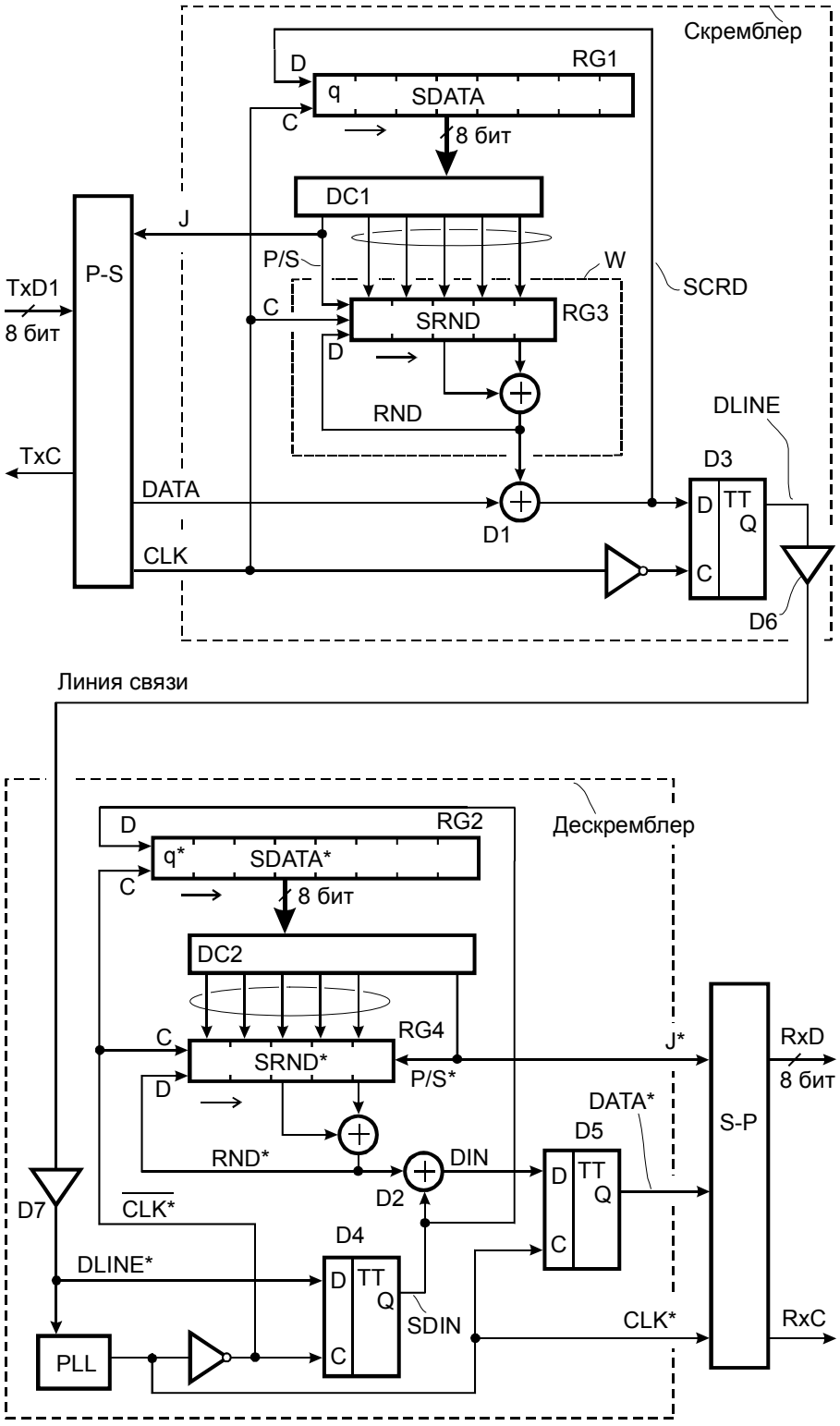
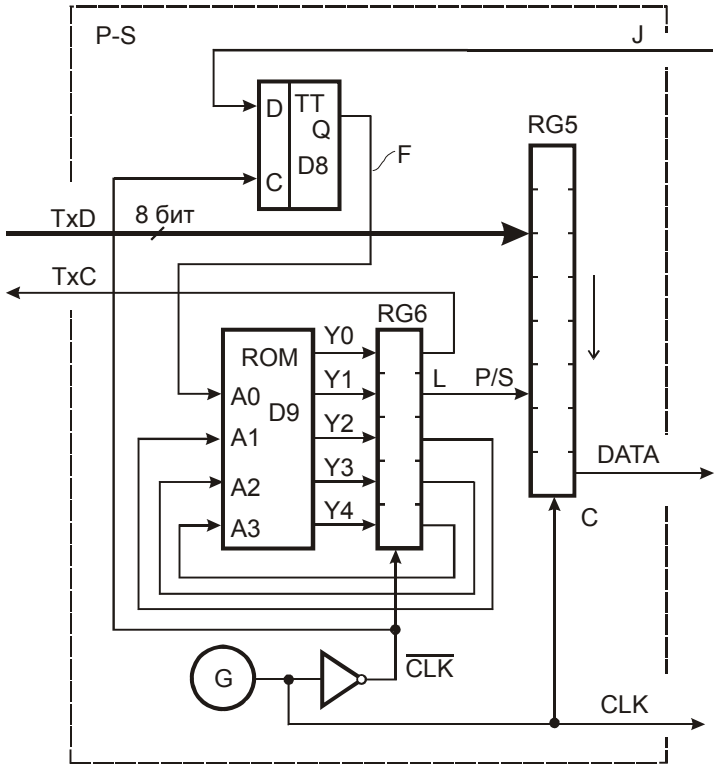


Рис. 4. Предлагаемая система передачи данных

Генератор W псевдослучайной последовательности битов скремблера выполнен на основе сдвигового регистра RG3. Аналогичный генератор на основе регистра RG4 содержится в дескремблере. Регистры RG3 и RG4 предназначены для временного хранения псевдослучайных кодов SRND и SRND*. В установившемся режиме эти коды одинаковы (совпадают с точностью до задержки передачи). Прием очередного бита в регистр с входа D происходит по положительному фронту сигнала на синхронизирующем входе C при условии, что на его управляющем входе P/S (P/S*), задающем режим параллельного или последовательного приема данных, присутствует сигнал лог. 0. Одновременно с приёмом очередного бита с входа D происходит сдвиг ранее хранимого кода на один разряд вправо.

Если на управляющем входе P/S (P/S*) регистра присутствует лог. 1, то по положительному фронту сигнала на синхронизирующем входе C в регистр принимается параллельный код с группы выходов дешифратора DC1 (DC2). В данном примере построения устройства разрядность регистров RG3 и RG4 выбрана равной пяти, хотя она может быть большей или меньшей.

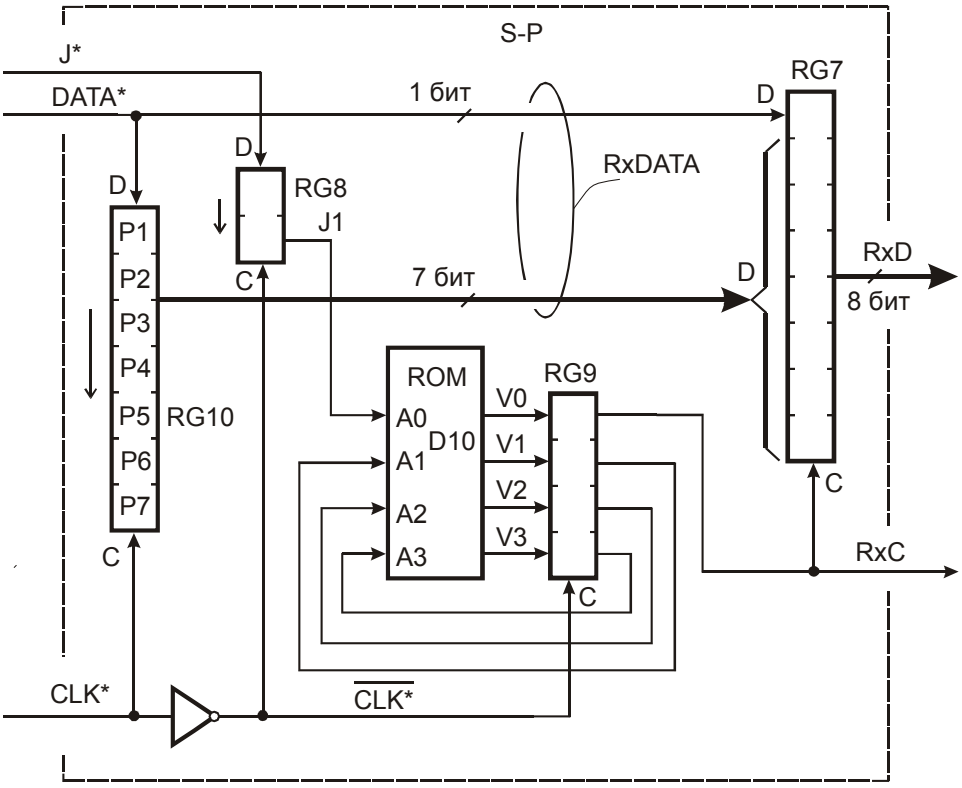


STATE	A0 A1 A2 A3	Y2 Y3 Y4	Y1	Y0
Z1	0000	001	1	1
Z2	0001	010	0	0
Z3	0010	011	0	0
Z4	0011	100	0	0
Z5	0100	101	0	0
Z6	0101	110	0	0
Z7	0110	111	0	0
Z8	0111	000	0	0
Z9	1000	001	1	1
Z10	1001	001	1	0
Z11	1010	001	1	0
Z12	1011	001	1	0
Z13	1100	001	1	0
Z14	1101	001	1	0
Z15	1110	001	1	0
Z16	1111	001	1	0

Рис. 5. Схема блока P-S преобразования параллельного кода в последовательный и кодировка ПЗУ D9 микропрограммного устройства управления

Элементы Исключающее ИЛИ D1 и D2 формируют скремблированный SCRD и дескремблированный DIN сигналы данных. Триггеры D-типа D3, D4 и D5 принимают биты данных с входа D по положительному фронту сигнала на входе синхронизации C. Триггеры D3 и D5 формируют выходные сигналы DLINE и DATA*, в которых на границах между битовыми интервалами сигнал может измениться только один раз, в то время как входные сигналы SCRD и DIN этих триггеров на границах между битовыми интервалами могут многократно изменяться из-за “гонок” сигналов. Триггер D4 в значительной степени устраняет джиттер входного сигнала (“дрожание” фронтов на границах между битовыми

интервалами) благодаря тому, что приём бита в этот триггер происходит в центре битового интервала, когда переходные процессы сигнала DLINE* уже закончились. Остаточный джиттер сигнала SDIN на выходе триггера D4 определяется неидеальностью сигнала CLK* на выходе генератора PLL с фазовой автоподстройкой частоты. Исходные состояния триггеров D3 – D5 произвольны.



STATE	A0 A1 A2 A3	V1 V2 V3	V0
Z1	0000	001	0
Z2	0001	010	0
Z3	0010	011	0
Z4	0011	100	0
Z5	0100	101	0
Z6	0101	110	0
Z7	0110	111	0
Z8	0111	000	1
Z9	1000	001	0
Z10	1001	001	0
Z11	1010	001	0
Z12	1011	001	0
Z13	1100	001	0
Z14	1101	001	0
Z15	1110	001	0
Z16	1111	001	0

Рис. 6. Схема блока S-P преобразования последовательного кода в параллельный и кодировка ПЗУ D10 микропрограммного устройства управления

Генератор PLL с фазовой автоподстройкой частоты может быть выполнен по одной из известных схем (см., например, [5]). Он предназначен для формирования высокостабильного синхросигнала CLK* на основе непрерывного слежения за входным сигналом DLINE*. Положительный фронт сигнала CLK* привязан к моментам изменения сигнала DLINE*, так что отрицательный фронт сигнала CLK* формируется в середине битового интервала сигнала DLINE*, что соответствует его установившемуся значению.

Благодаря достаточной инерционности генератора PLL сигнал CLK* практически нечувствителен к джиттеру сигнала DLINE* и иным его кратковременным искажениям, вызванным помехами в линии связи. Такое использование генератора с фазовой автоподстройкой частоты в телекоммуникационных системах является общепринятым и далее не детализируется.

Дешифратор DC1 (DC2) предназначен для выделения в потоке скремблированных данных, проходящем через скользящее окно – сдвиговый регистр RG1 (RG2), определенных кодов $CODE_1$, $CODE_2$, …, $CODE_K$. При обнаружении дешифратором DC1 (DC2) указанных кодов, на его выходах, обведённых на рис. 4 овалом, формируется соответствующий V-разрядный код $LOAD_1$, $LOAD_2$, …, $LOAD_K$ для последующей параллельной загрузки сдвигового регистра RG3 (RG4). В данном примере K = 4, V = 5 (подробнее – см. [1]) . При обнаружении любого кода $CODE_1$, $CODE_2$, …, $CODE_K$ дешифратор DC1 (DC2) формирует также единичный сигнал на входе P/S (P/S*) управления режимом работы регистра RG3 (RG4), подготавливая его к параллельному приёму данных по положительному фронту очередного синхроимпульса на входе C.

Усилитель D6 (D7) предназначен для передачи (приёма) скремблированного сигнала данных в линию (из линии). Параметры усилителей определяются типом линии связи, которая в наиболее простом варианте может быть выполнена в виде витой пары проводов, коаксиального или оптоволоконного кабеля. Линия связи может содержать последовательно включённые ретрансляторы, в которых могут использоваться блоки буферной памяти. Поэтому задержка прохождения сигнала между аппаратурой передачи и приёма данных может быть значительной и заранее не известной (но постоянной).

Генератор G синхросигналов, размещённый в блоке P-S (рис. 5), задаёт темп работы всей системы передачи данных. На выходе генератора формируется непрерывная последовательность импульсов со скважностью, равной двум. Байты данных TxD, представленные параллельным кодом, поступают на входы системы в ответ на положительные фронты сигнала TxC. Байт остаётся неизменным вплоть до формирования следующего положительного фронта сигнала TxC. Байт TxD записывается в регистр RG5 по положительному фронту сигнала C на его входе синхронизации при наличия сигнала L = 1 на его управляющем входе P/S. При L = 0 по положительным фронтам сигнала CLK в регистре RG5 происходит сдвиг данных на один разряд вниз.

В отсутствие импульса J блок P-S преобразует поток байтов TxD в равномерный поток битов DATA. Сигнал J = 1 вызывает коррекцию границ байтов в битовом потоке, если новые границы не совпадают со старыми. Этот сигнал временно запоминается в триггере D8 и поступает в микропрограммное устройство управления, выполненное на основе постоянного запоминающего устройства (ПЗУ) D9 и выходного параллельного регистра RG6. Сигнал J = 1 в необходимых случаях вызывает приостановку передачи текущего байта, его повторный параллельный приём с входов TxD (где он остаётся неизменным) и повторную последовательную выдачу через регистр RG5. Кодировка ПЗУ D9 (микропрограмма) и последовательности переходов по ней представлены в нижней части рис. 5.

В блоке S-P преобразования последовательного кода в параллельный (рис. 6) сдвиговый регистр RG7 формирует поток выходных байтов RxD, сдвиговый регистр RG8 выполняет функции элемента задержки для выравнивания фаз сигналов J* и DATA*. Параллельные данные RxDATA записываются в регистр RG7 по положительным фронтам сигнала RxC.

В отсутствие импульса J* блок S-P преобразует поток битов DATA* в равномерный поток байтов RxD. Сигнал J* = 1 задерживается сдвиговым регистром RG8 и поступает в микропрограммное устройство управления, выполненное на основе ПЗУ D10 и параллельного регистра RG9. Сигнал J* = 1 в необходимых случаях (когда новые границы не совпадают со старыми) вызывает повторное формирование текущего байта. Сигнал RxC при этом формируется только один раз, когда повторно сформированный байт RxDATA

подготовлен к записи в регистр RG7. Кодировка ПЗУ D10 (микропрограмма) и последовательности переходов по ней представлены в нижней части рис. 6.

Далее рассмотрена работа системы в целом.

В установившемся режиме, когда J = J* = 0, а синхронизация между передающей и приёмной аппаратурой ранее достигнута, поток байтов с входов системы передаётся по линии связи в виде последовательного потока битов и затем вновь преобразуется в поток байтов на её выходах. При этом благодаря ранее достигнутой синхронизации блок S-P (точнее, его микропрограмма) “знает” положение границ между байтами в битовом потоке данных, передаваемых по линии связи. Это позволяет правильно восстанавливать байты.

Блок P-S преобразования параллельного кода в последовательный работает по циклической микропрограмме (рис. 5), в соответствии с которой осуществляется следующая последовательность переходов между состояниями: Z1 – Z2 – Z3– … – Z8 – Z1 – Z2 и т. д. В ходе выполнения этой микропрограммы в каждом временном интервале из восьми тактов (точнее, в такте, соответствующем состоянию Z1) на выходах ПЗУ D9 формируются сигналы Y0 и Y1, которые в начале следующего такта переписываются в регистр RG6 (см. диаграммы сигналов TxC и L на рис. 7).

По положительному фронту сигнала TxC источник данных (на рис. 4, 5 не показан) присылает на входы системы очередной байт TxD. При L = 1 по положительному фронту сигнала CLK байт TxD принимается в регистр RG5, поэтому в интервале T1 – T2 (рис. 7) сигнал DATA отражает состояние нулевого разряда вновь принятого в этот регистр байта (разряды байтов условно пронумерованы с нулевого по седьмой). После скремблирования элементом D1 и прохождения через триггер D3 нулевой бит передается в линию связи. При L = 0 из регистра RG5 последовательно выдаются биты 1 – 7 текущего байта, затем при L = 1 в этот регистр принимается следующий байт и т. д.

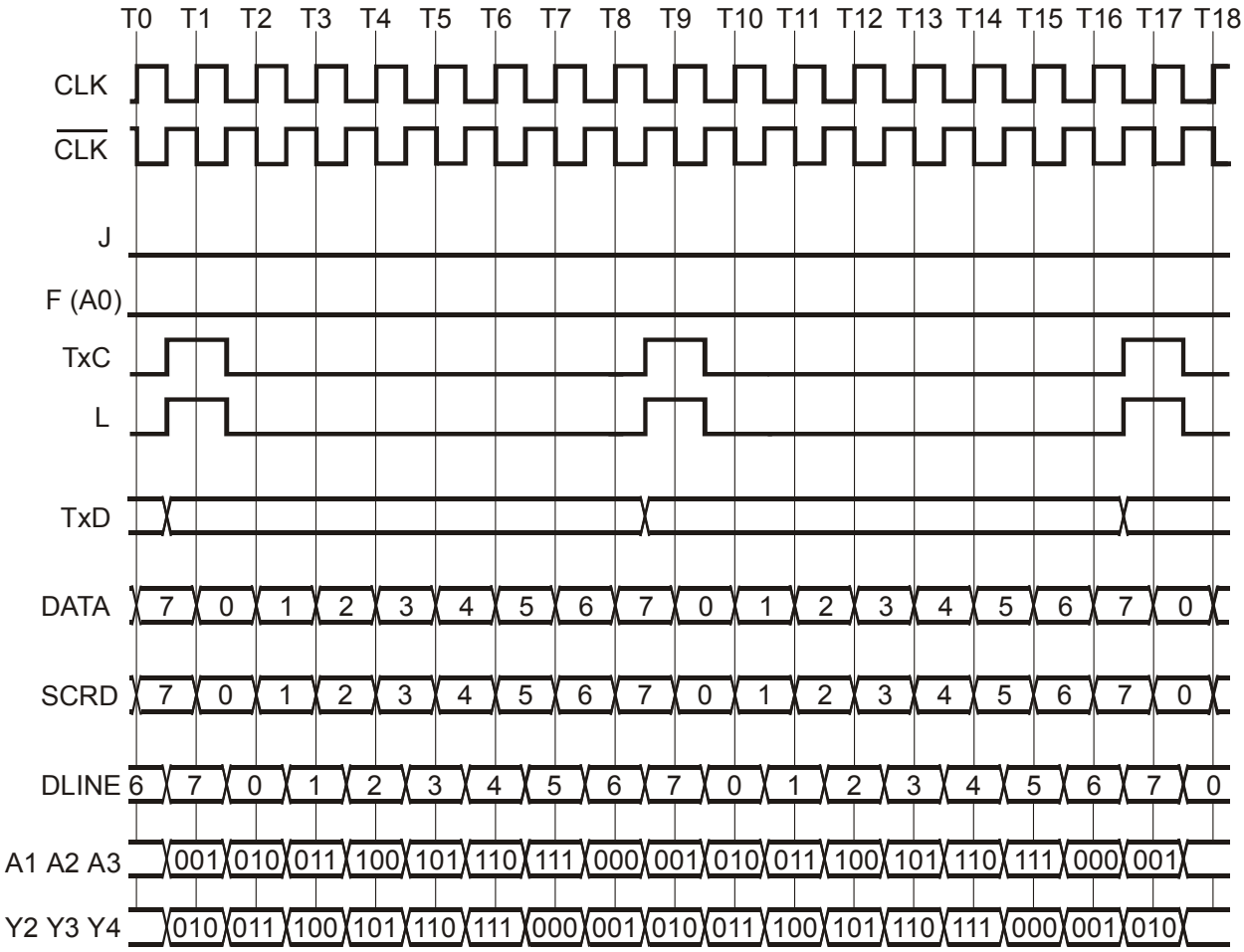


Рис. 7. Временные диаграммы работы блока P-S в отсутствие импульсов J

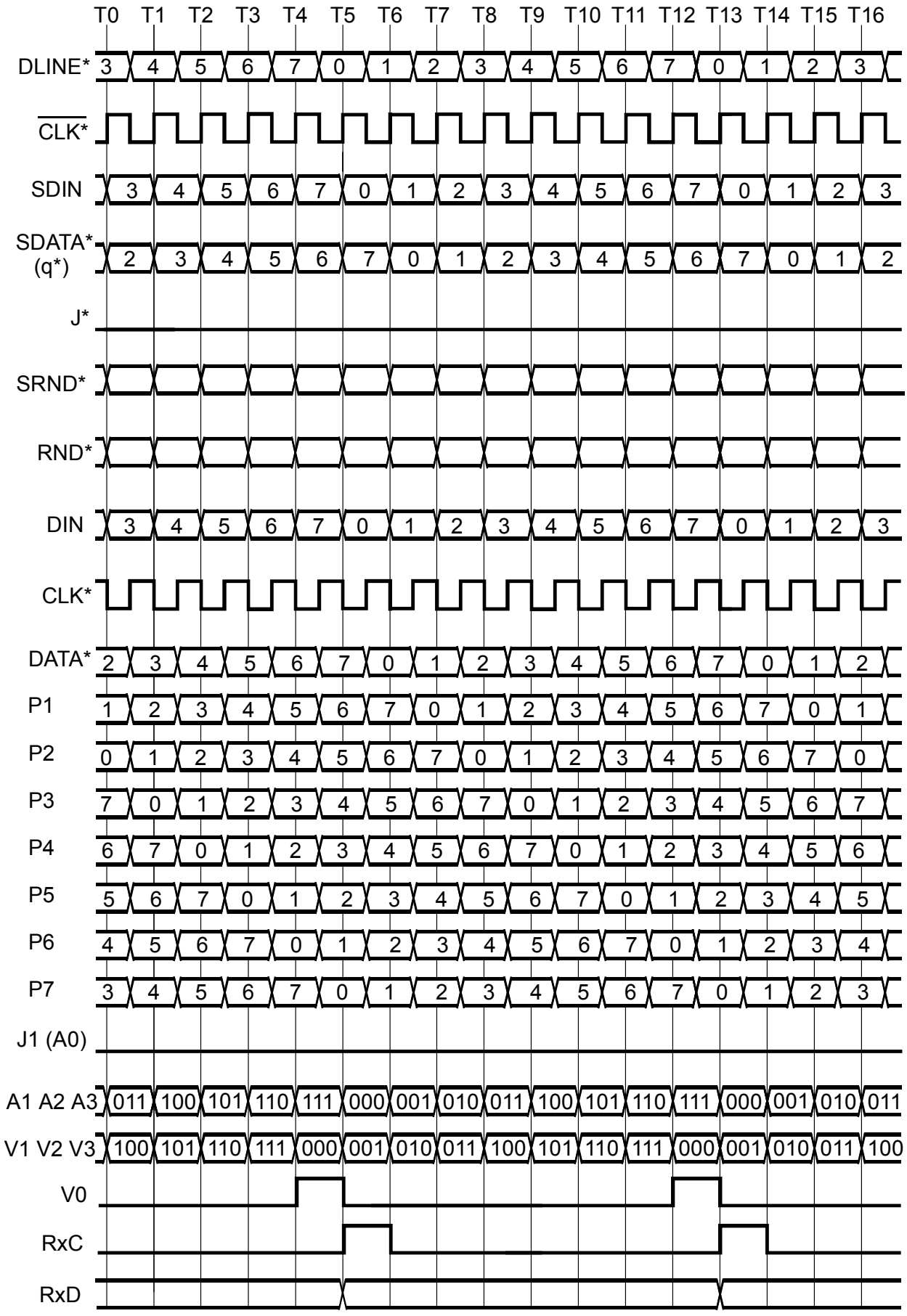


Рис. 8. Временные диаграммы работы блока S-P в отсутствие импульсов J*

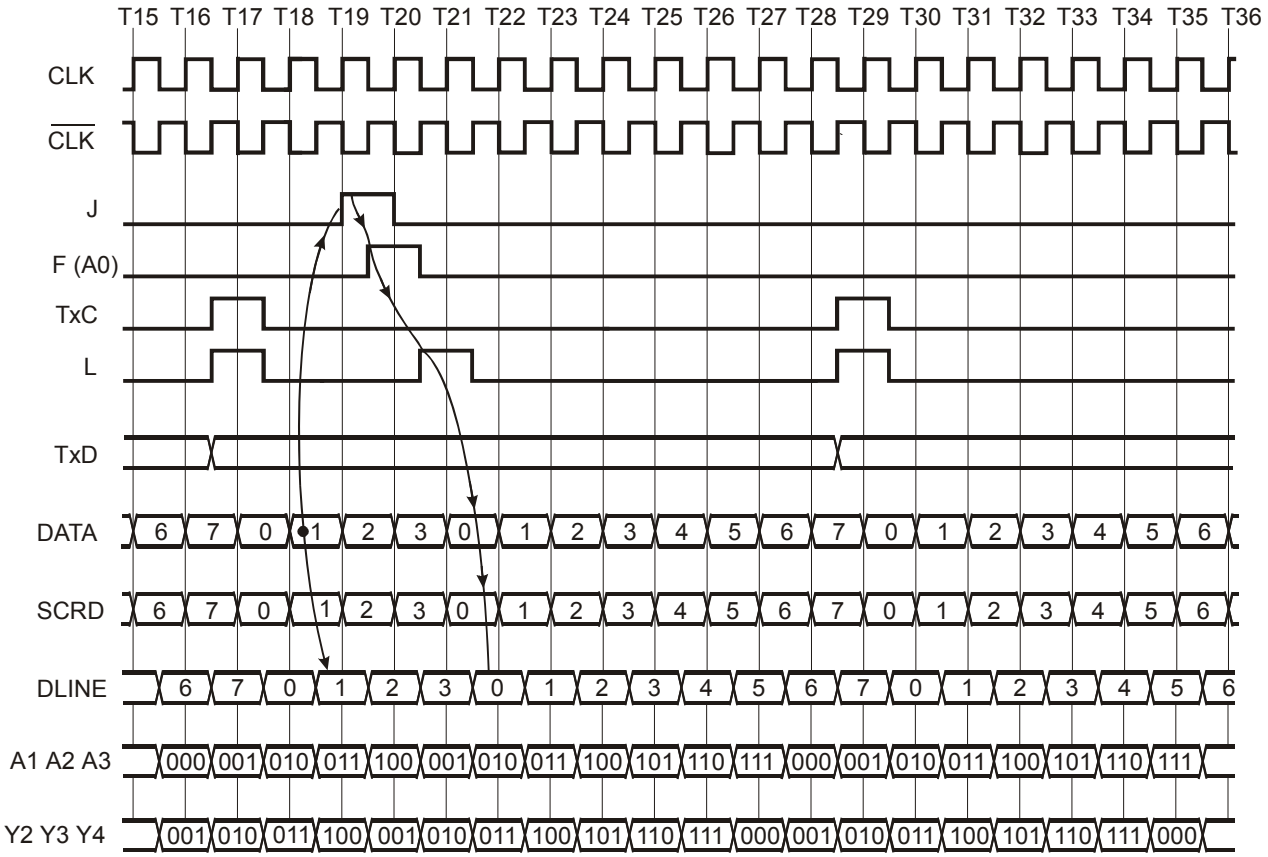


Рис. 9. Временные диаграммы работы блока P-S при наличии сигнала J

Блок S-P преобразования последовательного кода в параллельный также работает по циклической микропрограмме (рис. 6), в соответствии с которой осуществляется следующая последовательность переходов между состояниями: Z1 – Z2 – Z3– … – Z8 – Z1 – Z2 и т. д. В ходе выполнения этой микропрограммы в каждом временном интервале из восьми тактов (точнее, в такте, соответствующем состоянию Z8) на выходе ПЗУ D10 формируется сигнал V0 = 1, который в начале следующего такта переписывается в регистр RG9 и в виде сигнала RxC поступает на выход системы (см. диаграммы на рис. 8). К моментам формирования положительных фронтов сигнала RxC на входах данных регистра RG7 формируется очередной байт, который запоминается в этом регистре.

В режиме коррекции границ между байтами используются сигналы синхронизации J и J*. Эти сигналы, как уже отмечалось, формируются в заранее не известные (случайные) моменты времени в результате обнаружения в скремблированном потоке битов некоторых заданных кодов $CODE_1$, $CODE_2$, …, $CODE_K$.

Сигнал J = 1 задерживается триггером D8 на половину такта и преобразуется в сигнал F (см. диаграммы на рис. 9), который в качестве старшего разряда адреса поступает на вход ПЗУ D9 и вызывает переход в нижнюю половину таблицы состояний (рис. 5, состояние Z13), а затем в состояния Z2, Z3, …, Z8, Z1, Z2 и т. д. В результате в момент T21 (рис. 9) происходит повторный приём текущего байта в регистр RG5 и затем его последовательная выдача в линию связи. Для последующего распознавания приёмной аппаратурой факта изменения границ между байтами существенно, что интервал времени между битом, породившим сигнал J коррекции (в данном примере – битом с номером 1, который помечен точкой на диаграмме сигнала DATA, рис. 9), и повторно переданным нулевым битом всегда составляет два такта синхросигнала CLK. Так как в состоянии Z9 на выходах ПЗУ D9 формируются те же сигналы, что и в состоянии Z1, коррекция границ не осуществляется, когда новые границы соответствуют старым.

Блок S-P преобразования последовательного кода в параллельный принимает сигнал J* = 1 и задерживает его на два такта (см. диаграммы на рис. 10). Микропрограмма управ-

ления этим блоком (рис. 6) реагирует на поступление сигнала A0 = 1 переходом в одно из состояний Z9 – Z16, в данном примере – в состояние Z13. Далее начинается повторная последовательная загрузка байта в сдвиговый регистр RG10, которая завершается на половину такта раньше момента T25 формирования положительного фронта сигнала RxC. В момент T25 сформированный в новых временных границах байт записывается в регистр RG7.

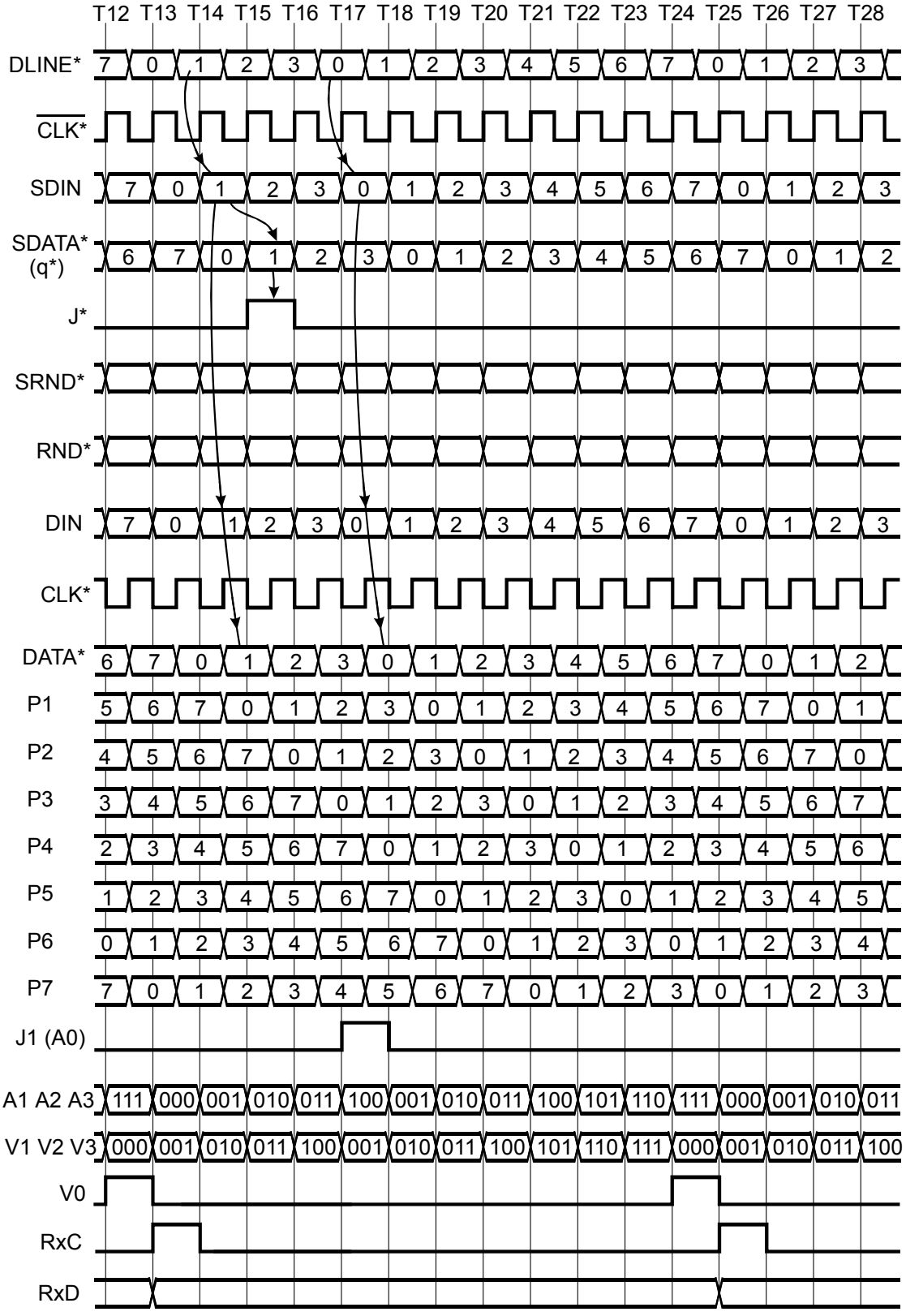


Рис. 10. Временные диаграммы работы блока S-P при наличии сигнала J*

После этого блоки P-S и S-P работают в новой системе отсчёта времени вплоть до момента получения очередной пары корректирующих импульсов J и J*, если они требуют смещения границ между байтами. Если коррекции не требуется (новая система отсчёта границ байтов совпадает со старой), то корректирующие импульсы игнорируются.

В исходном состоянии аппаратура приёма данных может быть не синхронизирована с передающей. После получения первого же импульса J* автоматически (без использования каких-либо программных средств) достигается кодовая синхронизация между дескремблером и скремблером (подробнее – см. [1]) и устанавливается единая система отсчёта границ байтов при их прохождении по линии связи.

Применение вероятностной синхронизации позволяет повысить скорость передачи данных благодаря двум факторам. Первый фактор состоит в исключении из потока данных относительно большого объёма служебной информации, предназначенной для синхронизации работы дескремблера со скремблером, а также в исключении из протоколов обмена соответствующих программных средств. Второй фактор – уменьшение объёма пересылаемой в потоке данных избыточной информации, обозначающей границы между байтами.

Потери времени из-за повторной передачи части байта составляют от нуля до семи битовых интервалов (в среднем, 3,5 интервала) на фоне периода между моментами коррекции, который в среднем может составлять, например, 100 мс. При скорости передачи 10 Гбит/с избыточность составляет $3{,}5 \times 10^{-9}$, т. е. один избыточный бит приходится на 286 млн “полезных” битов. Таким образом, предлагаемый способ синхронизации позволяет практически исключить избыточность из потока данных.

ЛИТЕРАТУРА:

1. *Б.В.Шевкопляс. Скремблирование передаваемых данных — Схемотехника, 2004, №12, с. 24 — 27, 2005, №1, с. 29 — 32, 2005, №2, с. 32 — 35, 2005, №3, с. 30 — 33. Электронная версия статьи*: *http://lit.lib.ru/s/shewkopljas_b_w/*
2. *Б.В.Шевкопляс. Вероятностная синхронизация в телекоммуникационных системах: вставка команд в поток данных без использования избыточных битов – Схемотехника, 2005, N5, с. 23 — 25, N6, с. 23 — 26. Электронная версия статьи: http://lit.lib.ru/s/shewkopljas_b_w/*
3. *Пат. заявка США US 2002 0191721 A. Электронная версия заявки: http://www.uspto.gov*
4. *Пат. США № 6.011.808. Электронная версия патента: http://www.uspto.gov*
5. *Пат. США № 6.215.835 B1. Электронная версия патента: http://www.uspto.gov*

Б. Шевкопляс