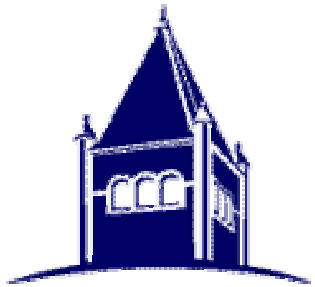


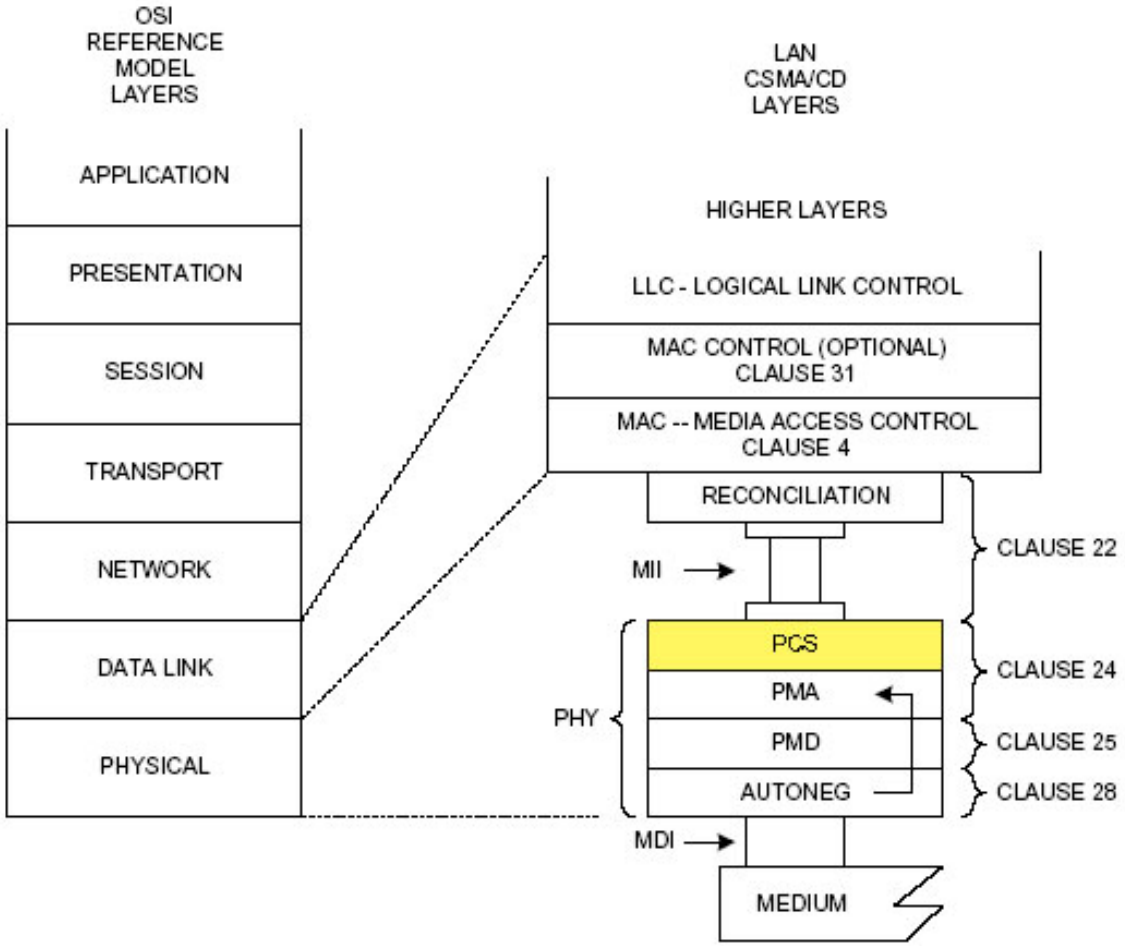
Clause 24

100BASE-X Physical Coding Sublayer (PCS)

Presentation Overview:

- Location in the OSI Stack
- Interface with Reconciliation sublayer
- Interface with Physical Medium Attachment (PMA) sublayer
- PCS Sublayer Functionality

PCS in the OSI Model



100BASE-TX

Interface with Reconciliation Sublayer

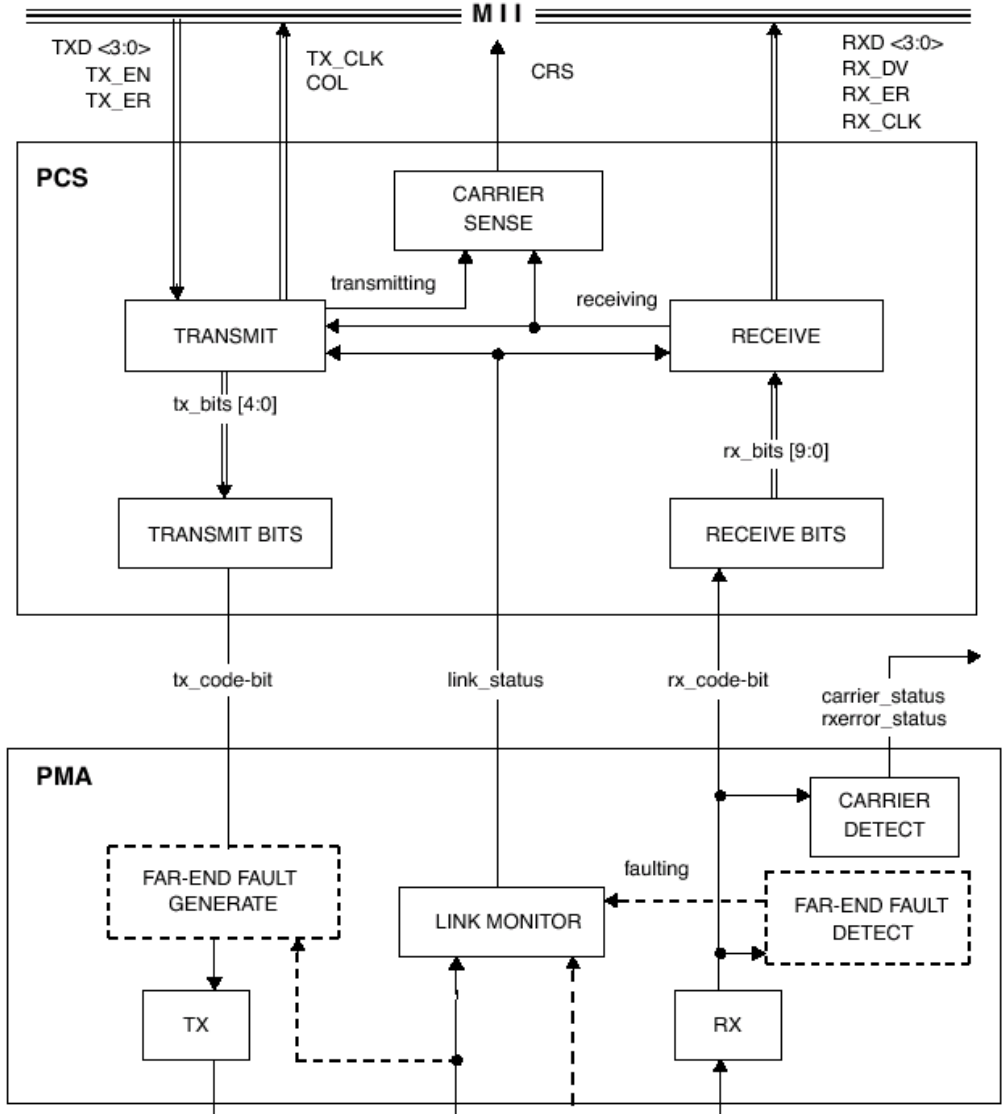
- The PCS sends and receives nibbles of data to the Reconciliation sublayer. It uses 16 signals to transfer and receive data, and to indicate collisions and carrier.
- RX signals
  - RXD<3:0> - 4 lines for received data nibbles
  - RX_ER – indicate a receive error
  - RX_DV – indicate the reception of valid data
  - RX_CLK – used as timing reference for transfer of RX signals

Interface with Reconciliation Sublayer

- TX signals
  - TXD<3:0> - 4 lines for transmitting data nibbles
  - TX_EN – indicate the beginning of frame transmission
  - TX_ER – used to force a transmission error by means of an invalid code group
  - TX_CLK – used as timing reference for transfer of TX signals
- Carrier Indication signals
  - CRS – indicates carrier activity on the receive channel
  - COL – indicates a collision on the medium

Interface with PMA Sublayer

- Serial Bit Stream
  - The PCS and PMA transfer code bits through serial bit streams.
- Control Status
  - The link_status indication is generated by the PMA and sent to the PCS. This indicates the integrity of the link (READY, OK, or FAIL).



PCS Sublayer Functions

- The 100BASE-X PCS provides the following services:
  - 4B/5B Encoding and Decoding
  - Carrier sense and Collision detection
  - Stream serialization to the underlying PMA
  - Mapping of the MII signals to the PMA

4B/5B Encoding

- The PCS layer receives 4-bit data nibbles from the MII. This results in a total of 2^4 different data symbols.
- The PCS needs additional control symbols to indicate idle, start and end of the data stream, and errors to the far-end station.
- The PCS would like sufficient transition density of the signal (varied transition of 1s and 0s) to aid in clock recovery at the far-end PMD.
- This is accomplished through 4B/5B encoding: mapping a 4-bit data nibble to a 5-bit code group.

4B/5B

- In the 5-bit world we now have 2^5 symbols referred to as "code-groups":
  - The 16 data symbols (0-9, A-F) still exist but with different bit representations. A 4-bit Data 0, 0000, gets translated into a 5-bit code-group /0/, 11110.
  - /J/ and /K/ define the start of a stream
  - /T/ and /R/ define the end of a stream
  - /I/ defines the idle code-group that separates streams
  - /H/ indicates an intentional error used in transmission
  - The remaining 10 code-groups are invalid and should never be intentionally transmitted (except for repeaters)

4B/5B Data Symbol List

Table 24–1—4B/5B code-groups

	PCS code-group [4:0] 4 3 2 1 0	Name	MII (TXD/RXD) <3:0> 3 2 1 0	Interpretation
DATA	1 1 1 1 0	0	0 0 0 0	Data 0
	0 1 0 0 1	1	0 0 0 1	Data 1
	1 0 1 0 0	2	0 0 1 0	Data 2
	1 0 1 0 1	3	0 0 1 1	Data 3
	0 1 0 1 0	4	0 1 0 0	Data 4
	0 1 0 1 1	5	0 1 0 1	Data 5
	0 1 1 1 0	6	0 1 1 0	Data 6
	0 1 1 1 1	7	0 1 1 1	Data 7
	1 0 0 1 0	8	1 0 0 0	Data 8
	1 0 0 1 1	9	1 0 0 1	Data 9
	1 0 1 1 0	A	1 0 1 0	Data A
	1 0 1 1 1	B	1 0 1 1	Data B
	1 1 0 1 0	C	1 1 0 0	Data C
	1 1 0 1 1	D	1 1 0 1	Data D
	1 1 1 0 0	E	1 1 1 0	Data E
	1 1 1 0 1	F	1 1 1 1	Data F

4B/5B Control Symbol List

	PCS code-group [4:0] 4 3 2 1 0	Name	MII (TXD/RXD) <3:0> 3 2 1 0	Interpretation
	1 1 1 1 1	I	undefined	IDLE; used as inter-stream fill code
CONTROL	1 1 0 0 0	J	0 1 0 1	Start-of-Stream Delimiter, Part 1 of 2; always used in pairs with K
	1 0 0 0 1	K	0 1 0 1	Start-of-Stream Delimiter, Part 2 of 2; always used in pairs with J
	0 1 1 0 1	T	undefined	End-of-Stream Delimiter, Part 1 of 2; always used in pairs with R
	0 0 1 1 1	R	undefined	End-of-Stream Delimiter, Part 2 of 2; always used in pairs with T

4B/5B Invalid Symbol List

	PCS code-group [4:0] 4 3 2 1 0	Name	MII (TXD/RXD) <3:0> 3 2 1 0	Interpretation
INVALID	0 0 1 0 0	H	Undefined	Transmit Error; used to force signaling errors
	0 0 0 0 0	V	Undefined	Invalid code
	0 0 0 0 1	V	Undefined	Invalid code
	0 0 0 1 0	V	Undefined	Invalid code
	0 0 0 1 1	V	Undefined	Invalid code
	0 0 1 0 1	V	Undefined	Invalid code
	0 0 1 1 0	V	Undefined	Invalid code
	0 1 0 0 0	V	Undefined	Invalid code
	0 1 1 0 0	V	Undefined	Invalid code
	1 0 0 0 0	V	Undefined	Invalid code
	1 1 0 0 1	V	Undefined	Invalid code

From 4B to 5B

- A few changes take place when converting from 4B to 5B
  - When TX_EN is asserted on the MII (to indicate the beginning of a MAC frame) the first two nibbles of data are replaced with the /J/ and /K/ code-groups. The first two nibbles of data should be the first two nibbles of the MAC preamble (0101 0101). The /J/K/ combination of code-groups is known as the Start of Stream Delimiter (SSD).
  - When TX_EN is de-asserted (to indicate the end of the MAC frame) the PCS transmits the /T/ and /R/ code-groups. The /T/R/ combination of signals is know as the End of Stream Delimiter (ESD).
  - After transmitting the ESD the PCS transmits the idle code-group

From 5B to 4B

- A few more changes take place when converting from 5B to 4B:
  - When the PCS receives the SSD (/J/K/) it replaces it with 0101 0101. This is interpreted as preamble by the MAC.
  - When the PCS receives the ESD (/T/R/) it de-asserts RX_DV to let higher layers know they are no longer receiving valid data.
  - When the PCS receives one of the invalid code-groups (/H/ or the 10 others) it signals an error via the RX_ER signal.

PCS Encapsulation

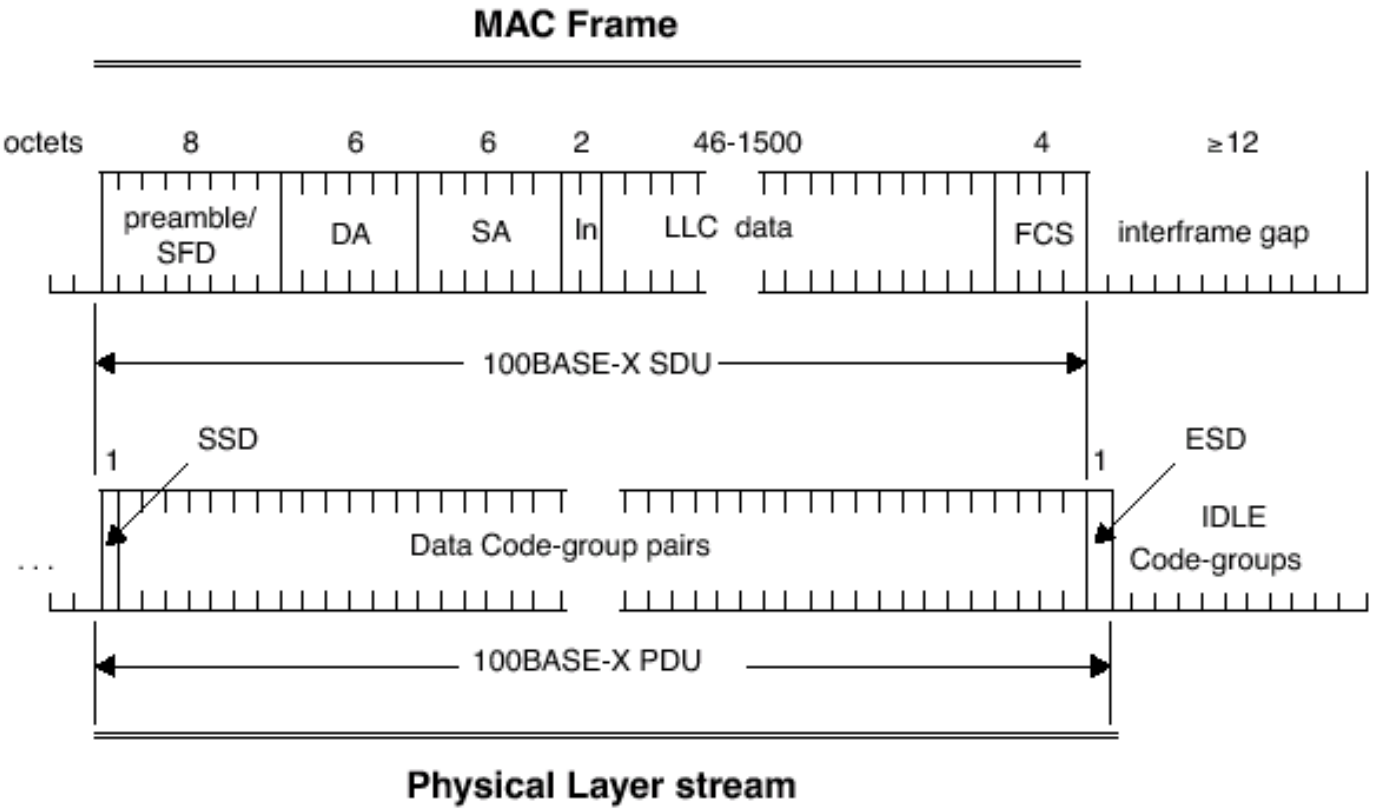


Figure 24–5—PCS encapsulation

4B/5B Summary

- Only the PCS knows about 5-bit code-groups (except for the PMA)
- /J/ and /K/ replace part of the MAC frame's preamble
- /T/ and /R/ replace part of the inter-frame gap
- /H/ is the only invalid code group that can intentionally be transmitted (except for repeaters)

Carrier and Collision Detection

- The PCS can detect activity on both the transmit and receive channels. If it detects only one channel has activity it signals CRS on the MII (meaning there is activity on the medium).
- If both channels show activity the PCS can signal a collision to higher layers via the MII's COL signal.
- This is only collision detection, not enforcement. The MAC layer enforces collisions in Half Duplex mode.
- The behavior of the COL signal is undefined when the Phy is in Full Duplex.

Stream Serialization

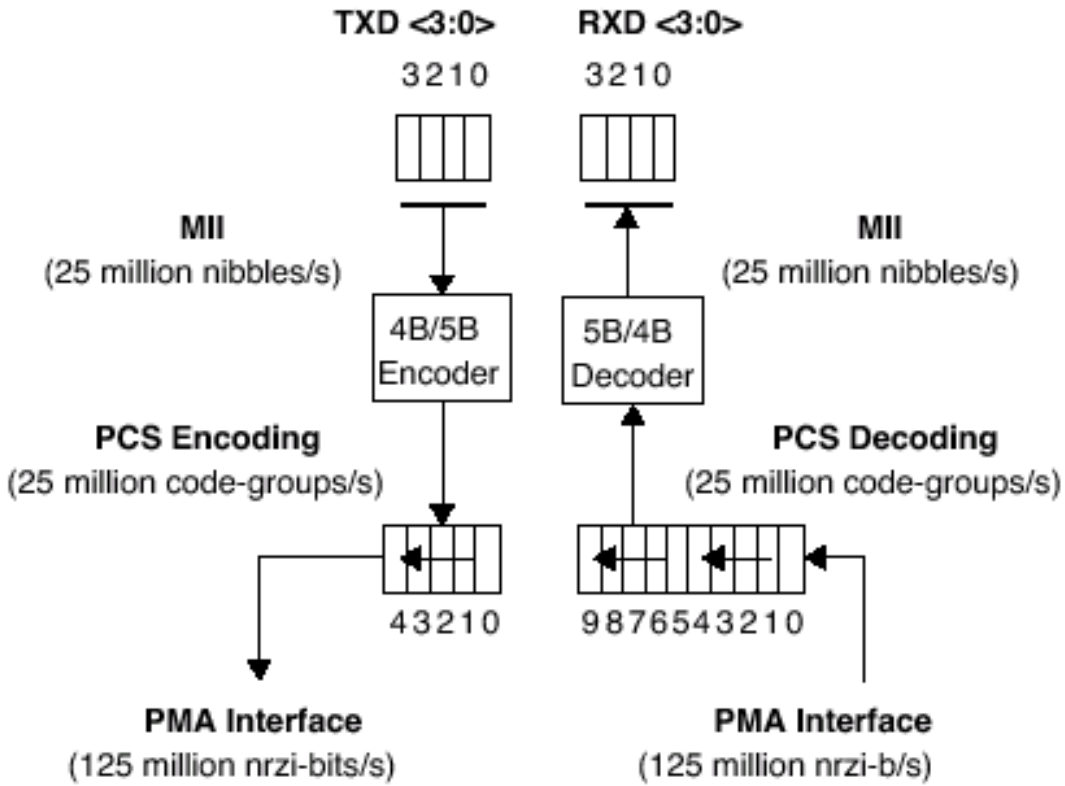


Figure 24–6—PCS reference diagram

PCS State Diagrams

- The following state diagrams define the transmit and receive functions of the PCS.
- The state diagrams are the final word on how the PCS should perform.
- We'll begin by showing a valid instance of transmission and reception, then show where the error cases occur.

- **Stream Transmission**
- Data code-groups are transmitted to the underlying PMA only when TX_EN is TRUE. Otherwise the Idle code-group (/I/) is transmitted.
- The SSD is transmitted during the first two received nibbles after TX_EN is TRUE (effectively ignoring the first two nibbles of data).
- The ESD is transmitted after TX_EN is FALSE.

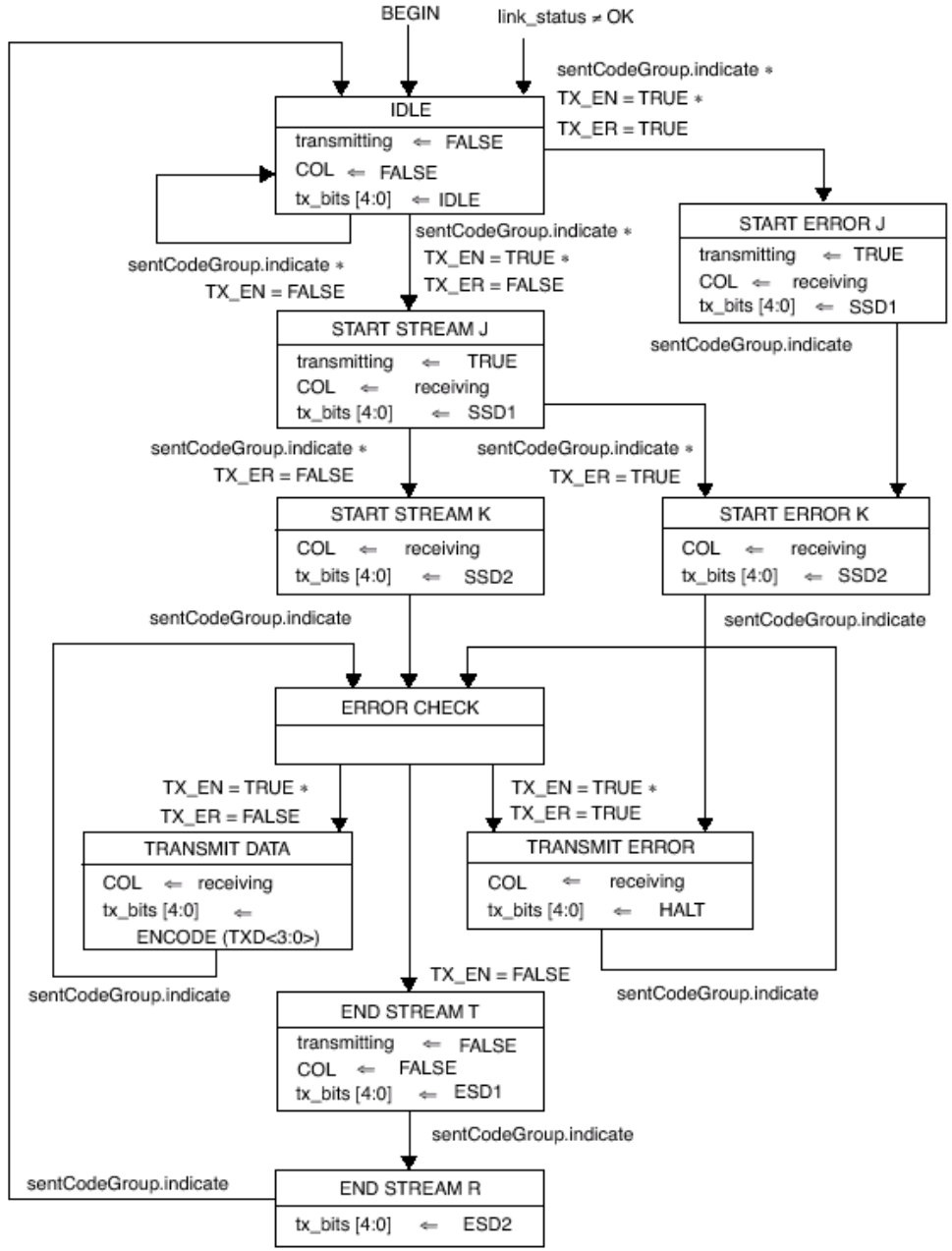


Figure 24–8—Transmit state diagram

IDLE

PCS initialization state.

transmitting=FALSE

•PCS is not transmitting

COL=FALSE

•There is no collision indicated because the PCS is not transmitting.

tx_bits[4:0]=IDLE

•Transmit the 5-bit IDLE code-group (11111)

Exit cases

sentCodeGroup.indicate and TX_EN=FALSE

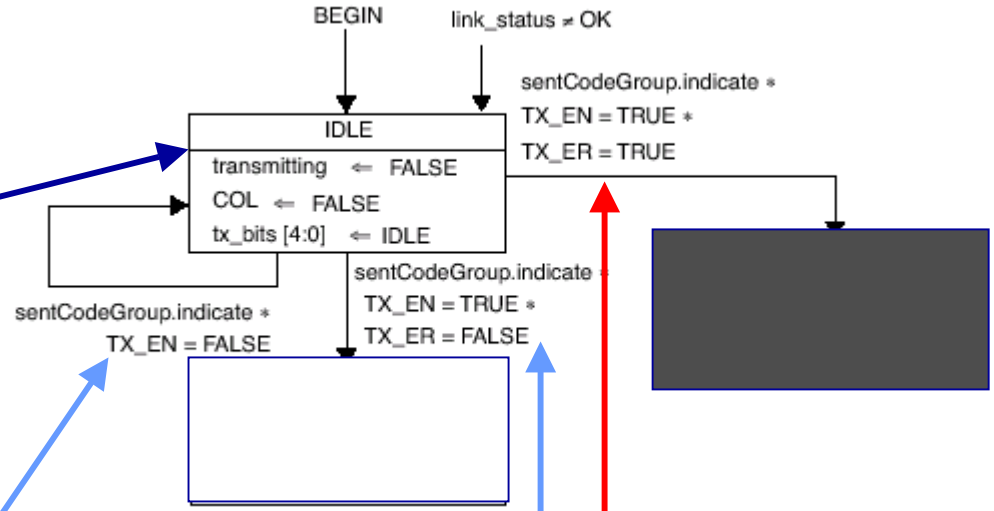
•Loop through the IDLE state getting data nibbles as long as there is no transmission

sentCodeGroup.indicate and TX_EN=TRUE and TX_ER=FALSE

•Exit the IDLE state when a transmission is started without error indication

sentCodeGroup.indicate and TX_EN=TRUE and TX_ER=TRUE

•Follow the error path of the state diagram when both TX_EN and TX_ER are TRUE



START STREAM J

Beginning of Start of Stream Delimiter (/J/K/) transmission.

transmitting=TRUE

•PCS is now transmitting

COL=receiving

•COL takes on the value of *receiving*, the indicator of PCS reception. If *receiving* is TRUE then COL is TRUE meaning a collision is occurring.

tx_bits[4:0]=SSD1

•Transmit SSD1, /J/ (11000)

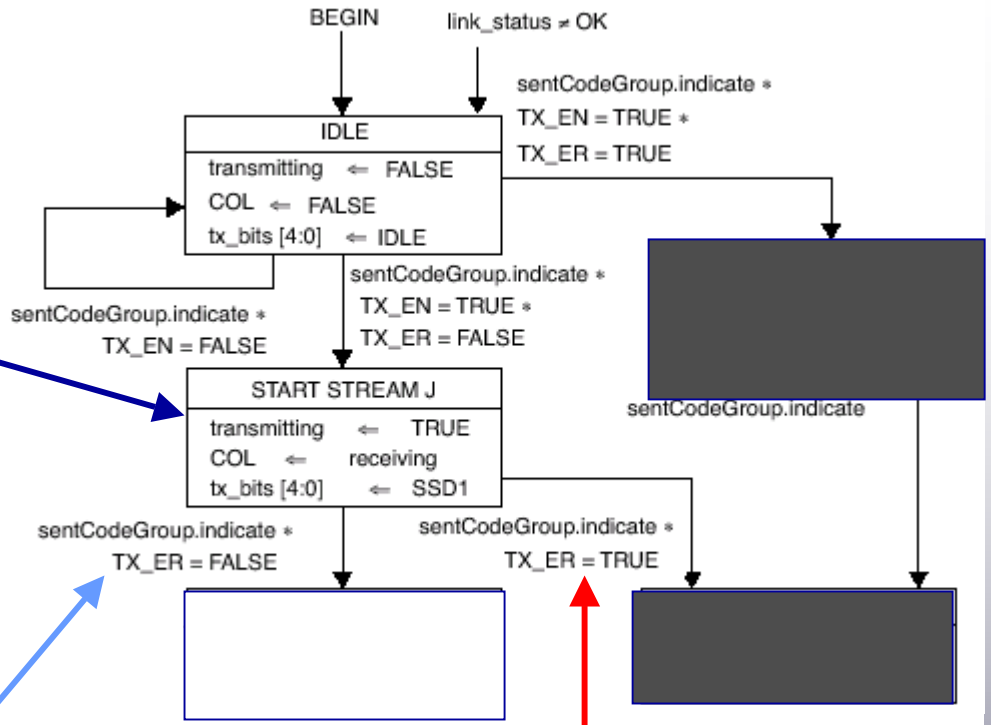
Exit cases

sentCodeGroup.indicate and TX_ER=FALSE

•Exit when a code group is available and an error is not forced

sentCodeGroup.indicate and TX_ER=TRUE

•Get on the error path when TX_ER is TRUE, but continue to transmit a valid SSD.



START STREAM K

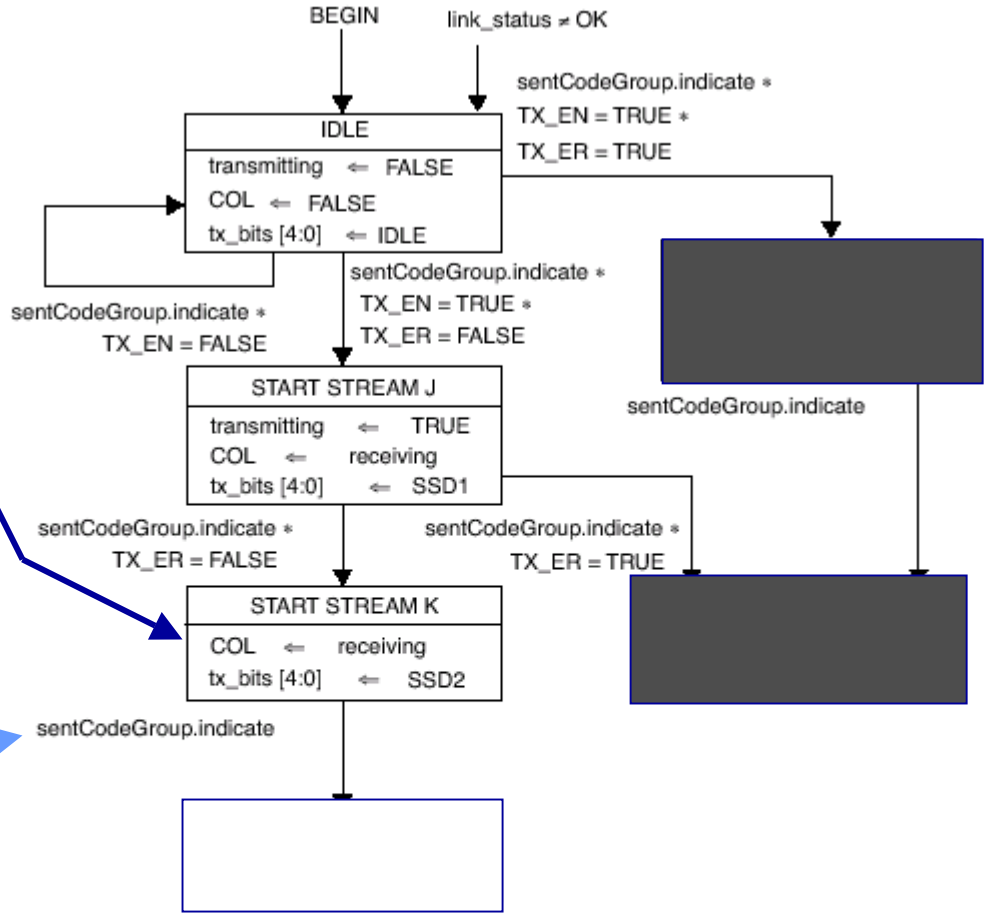
Second part of /J/K/ transmission.

COL=receiving

•COL takes on the value of *receiving* to indicate a collision or not.

tx_bits[4:0]=SSD2

•Transmit SSD2, /K/ (10001)



Exit cases

sentCodeGroup.indicate

•Exit to the ERROR CHECK state when a code group is available

ERROR CHECK

From this state the PCS loops through the transmission of valid data and the transmission of errors (/H/). When the transmission ends it exits to the end of stream path.

Exit cases

TX_EN=TRUE and TX_ER=FALSE

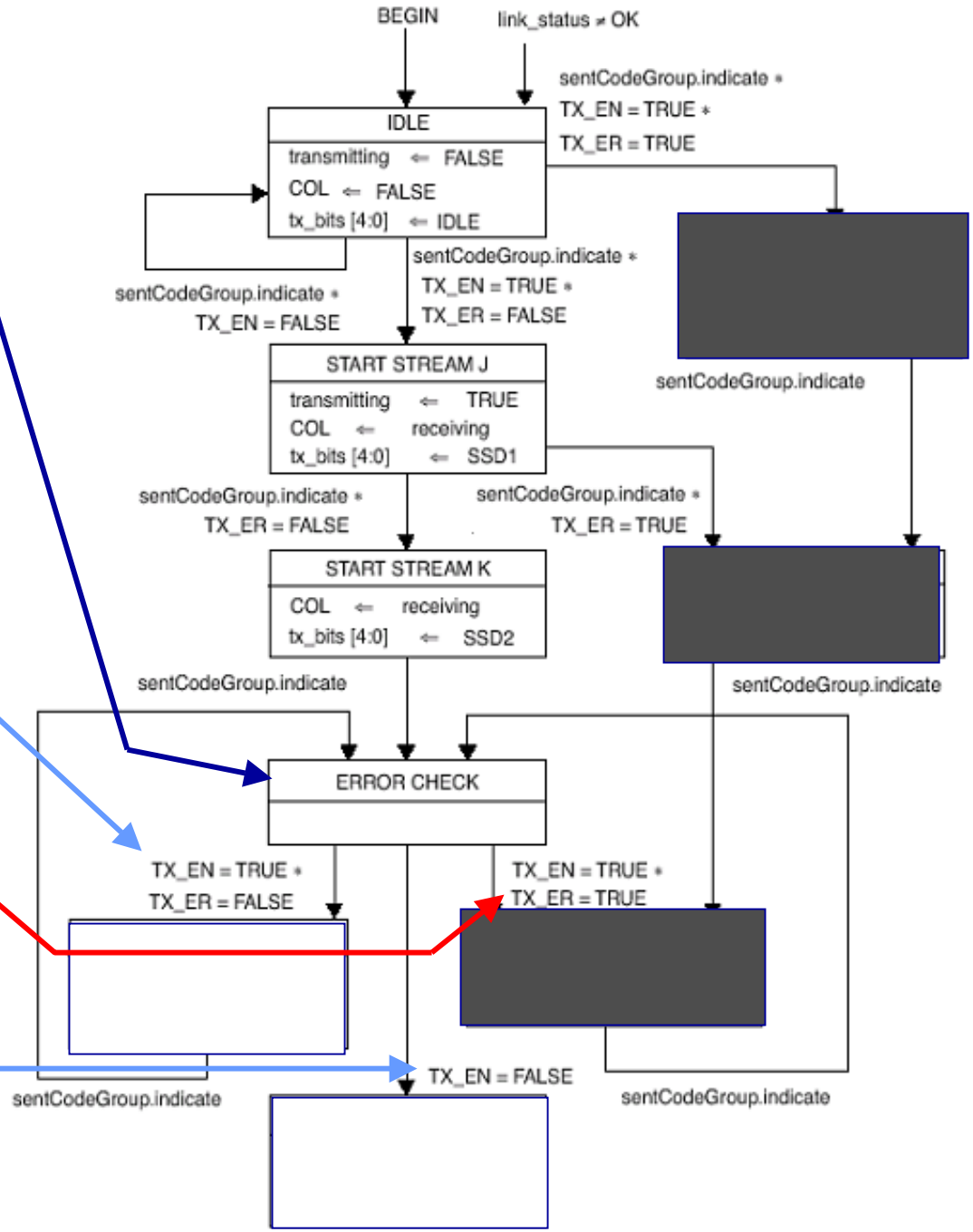
•Go to the TRANSMIT DATA state if transmission is enabled and transmit error is FALSE

TX_EN=TRUE and TX_ER=TRUE

•If an error is forced via TX_ER then the PCS enters the TRANSMIT ERROR state.

TX_EN=FALSE

•Exit the ERROR CHECK loop when the transmission is no longer enabled.



TRANSMIT DATA

Transmission of valid data symbols

COL=receiving

•COL takes on the value of *receiving* to indicate a collision or not.

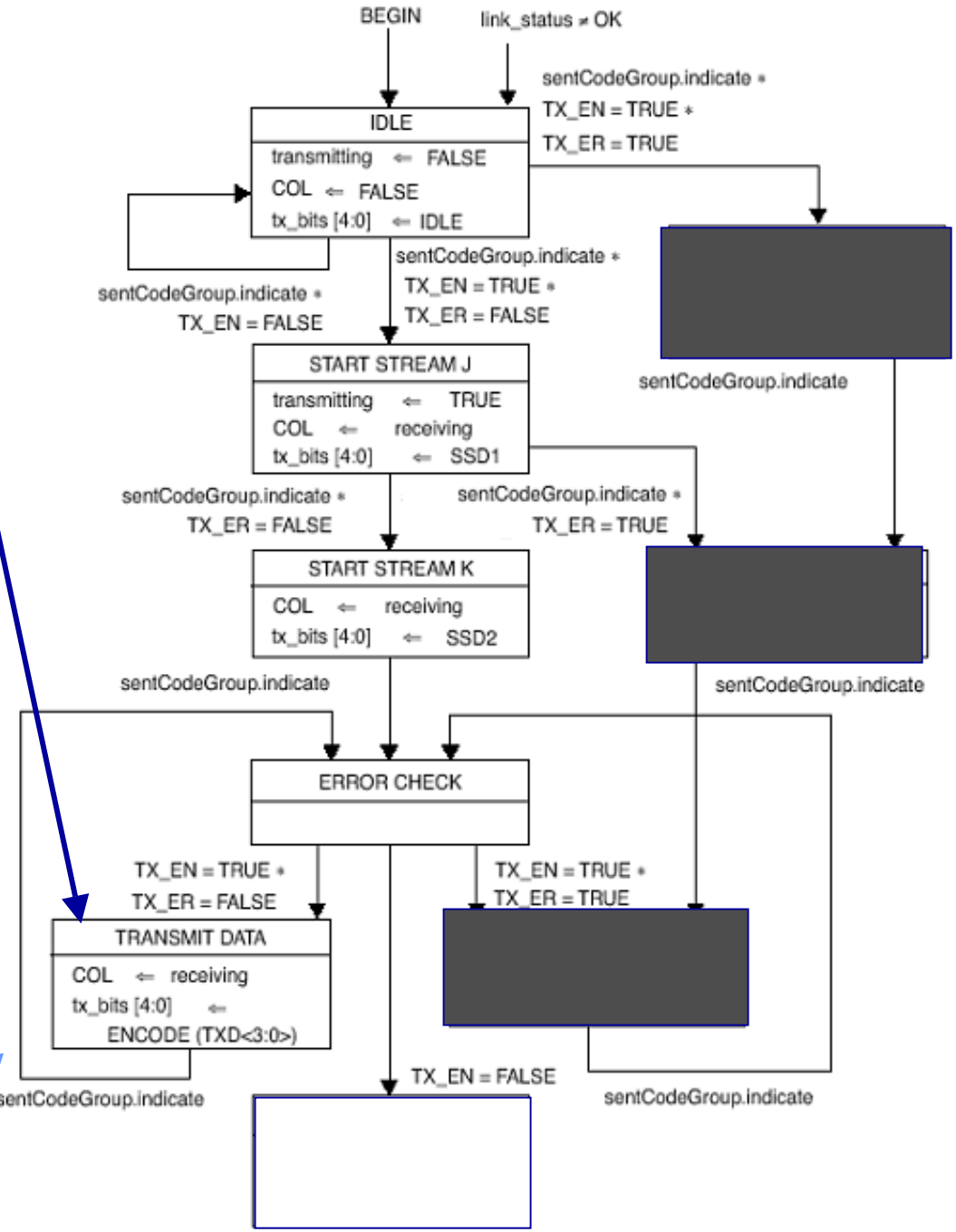
tx_bits[4:0]=ENCODE(TXD<3:0>)

•Encode the 4-bit nibble of data (from sentCodeGroup.indicate) into the corresponding 5-bit code-group

Exit cases

sentCodeGroup.indicate

•Exit to the ERROR CHECK state when a code group is available



END STREAM T

First part of /T/R/ transmission.

transmitting=FALSE

•PCS is no longer transmitting data nibbles from the MII

COL=FALSE

•Transmission is done so there is no more chance of a collision

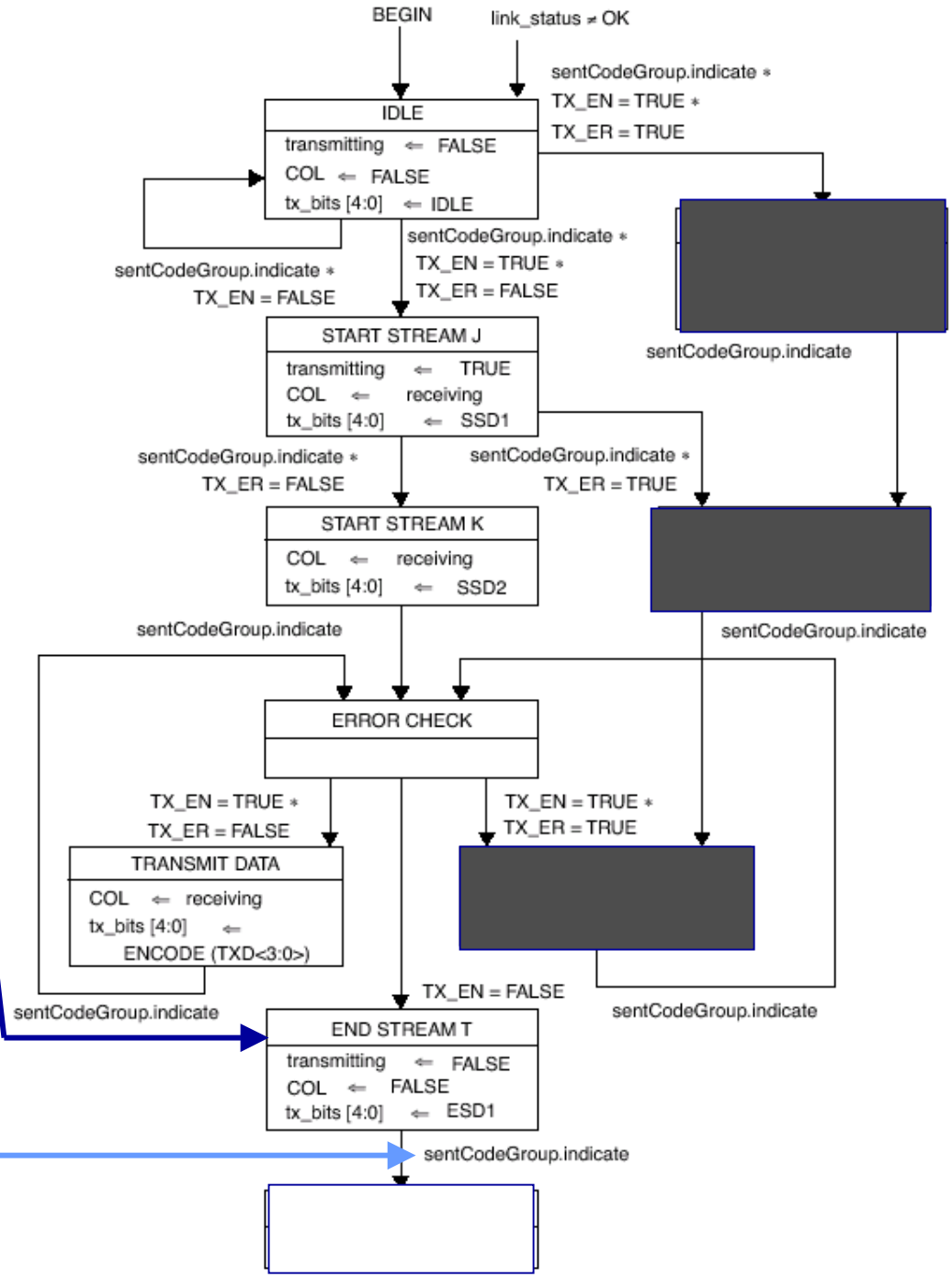
tx_bits[4:0]=ESD1

•Transmit ESD1, /T/ (01101). The data nibble from sentCodeGroup.indicate is ignored.

Exit cases

sentCodeGroup.indicate

•Exit to the END STREAM R state when a code group is available



END STREAM R

Second part of /T/R/ transmission.

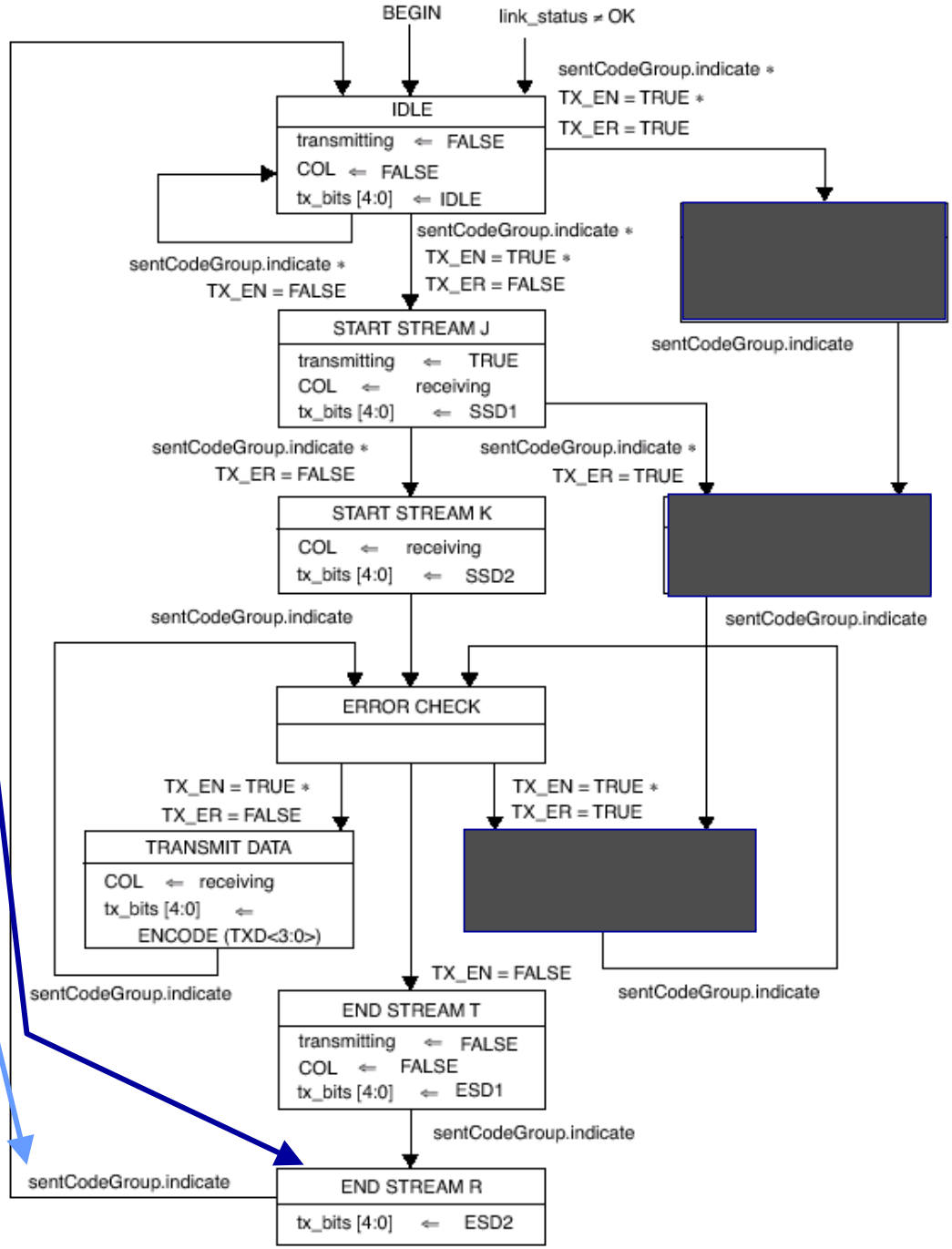
tx_bits[4:0]=ESD2

•Transmit ESD2, /R/ (00111). The data nibble from sentCodeGroup.indicate is ignored.

Exit cases

sentCodeGroup.indicate

•Exit to the IDLE state when a code group is available. This ends a stream transmission from the PCS.



- **Transmit Errors**
- The transmit error signal (TX_ER) can occur during /J/, /K/, or data (shown by the red arrows)
- Transmission errors should only occur when both TX_EN and TX_ER are TRUE.
- If TX_ER is TRUE during the transmission of the SSD (/J/K/), a valid SSD is transmitted anyway. At least one error code-group is transmitted afterwards.
- Transmission errors are created by sending the /H/ code-group.

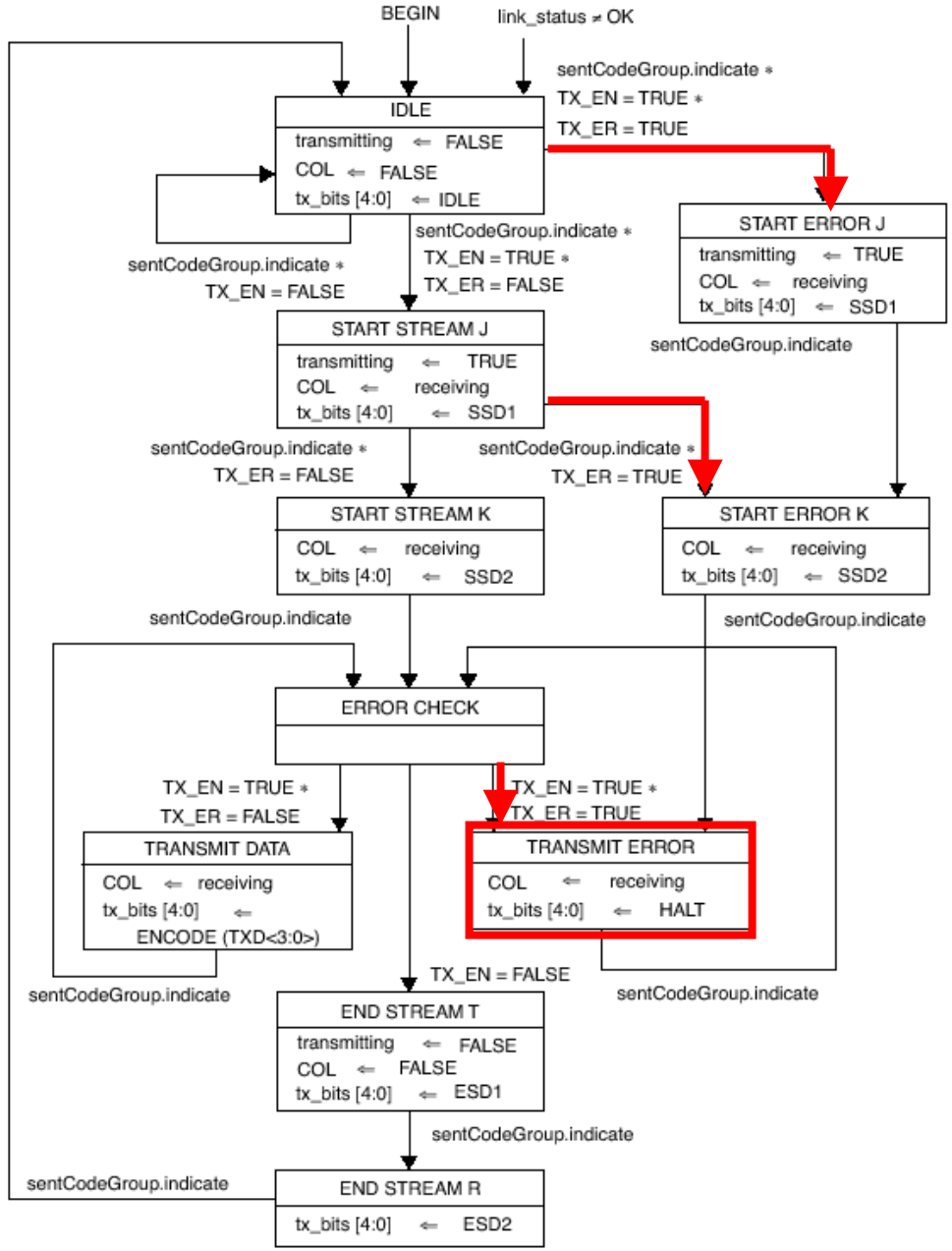


Figure 24–8—Transmit state diagram

IDLE Exit cases

sentCodeGroup.indicate and TX_EN=TRUE and TX_ER=TRUE

•When a transmission begins from the IDLE state and an error is forced the state diagram follows the error path to START ERROR J.

START ERROR J

Transmission of SSD when an error is indicated

transmitting=TRUE

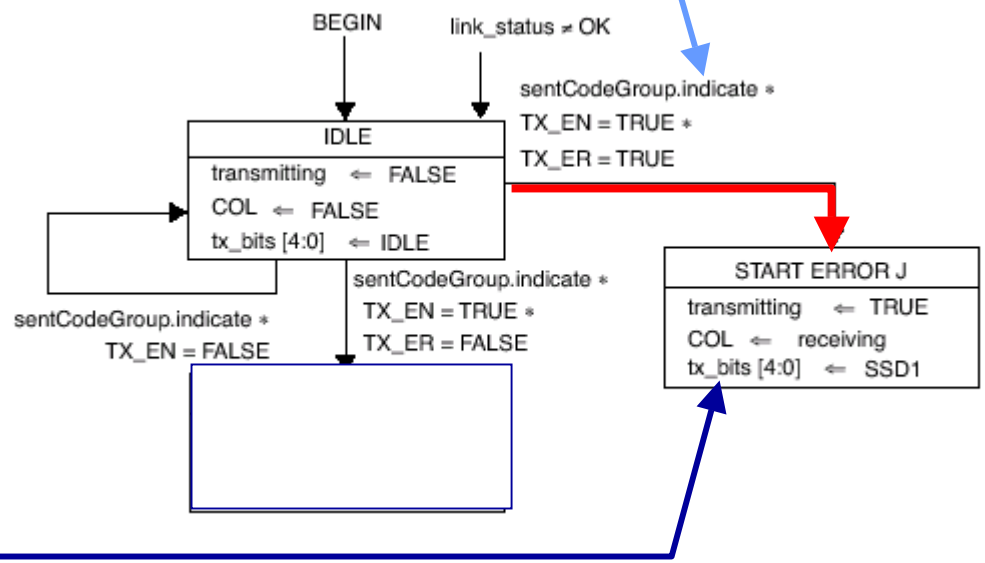
•PCS is transmitting

COL=receiving

•COL takes on the value of *receiving* to indicate a collision or not

tx_bits[4:0]=SSD1

•Transmit SSD1, /J/ (11000). Even though an error is indicated a valid SSD must be sent so the receiving station sees a valid Start of Stream Delimiter.



START STREAM J Exit cases

sentCodeGroup.indicate and TX_ER=TRUE

•If TX_ER is asserted after transmitting a valid /J/, the PCS enters the START ERROR K state. Otherwise it enters the START STREAM K state.

START ERROR K

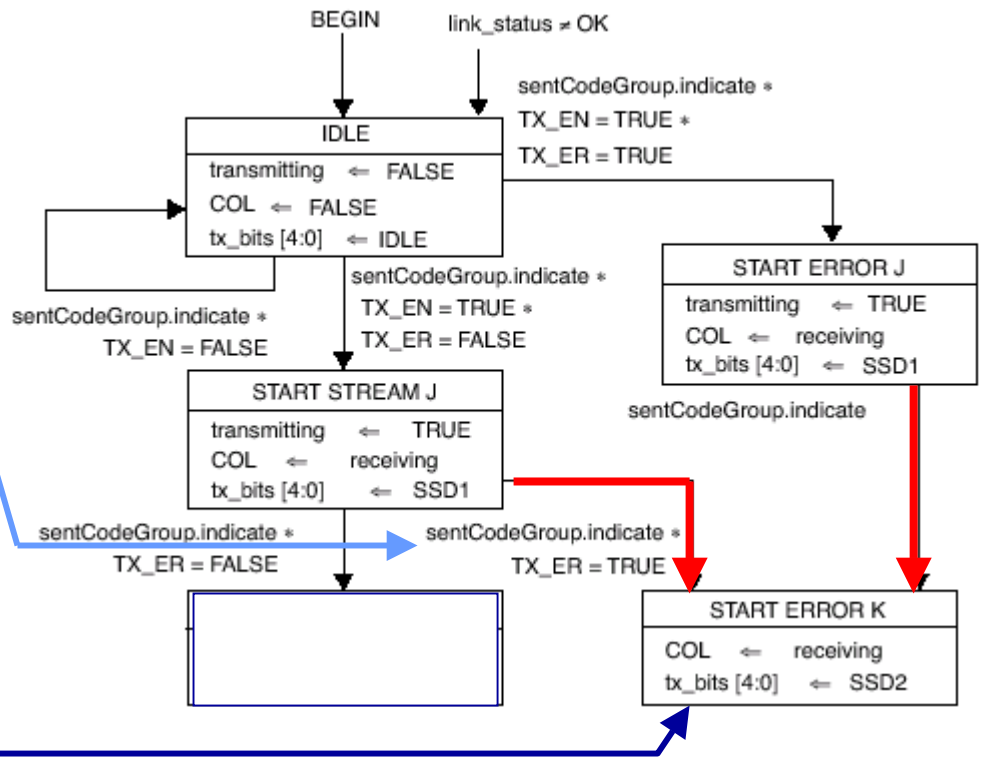
Transmission of SSD when an error is indicated

COL=receiving

•COL takes on the value of *receiving* to indicate a collision or not

tx_bits[4:0]=SSD2

•Transmit SSD2, /K/ (10001). Even though an error is indicated a valid SSD must be sent so the receiving station sees a valid Start of Stream Delimiter.



TRANSMIT ERROR

Transmission of the /H/ invalid code-group to force a transmit error

COL=receiving

•COL takes on the value of *receiving* to indicate a collision or not.

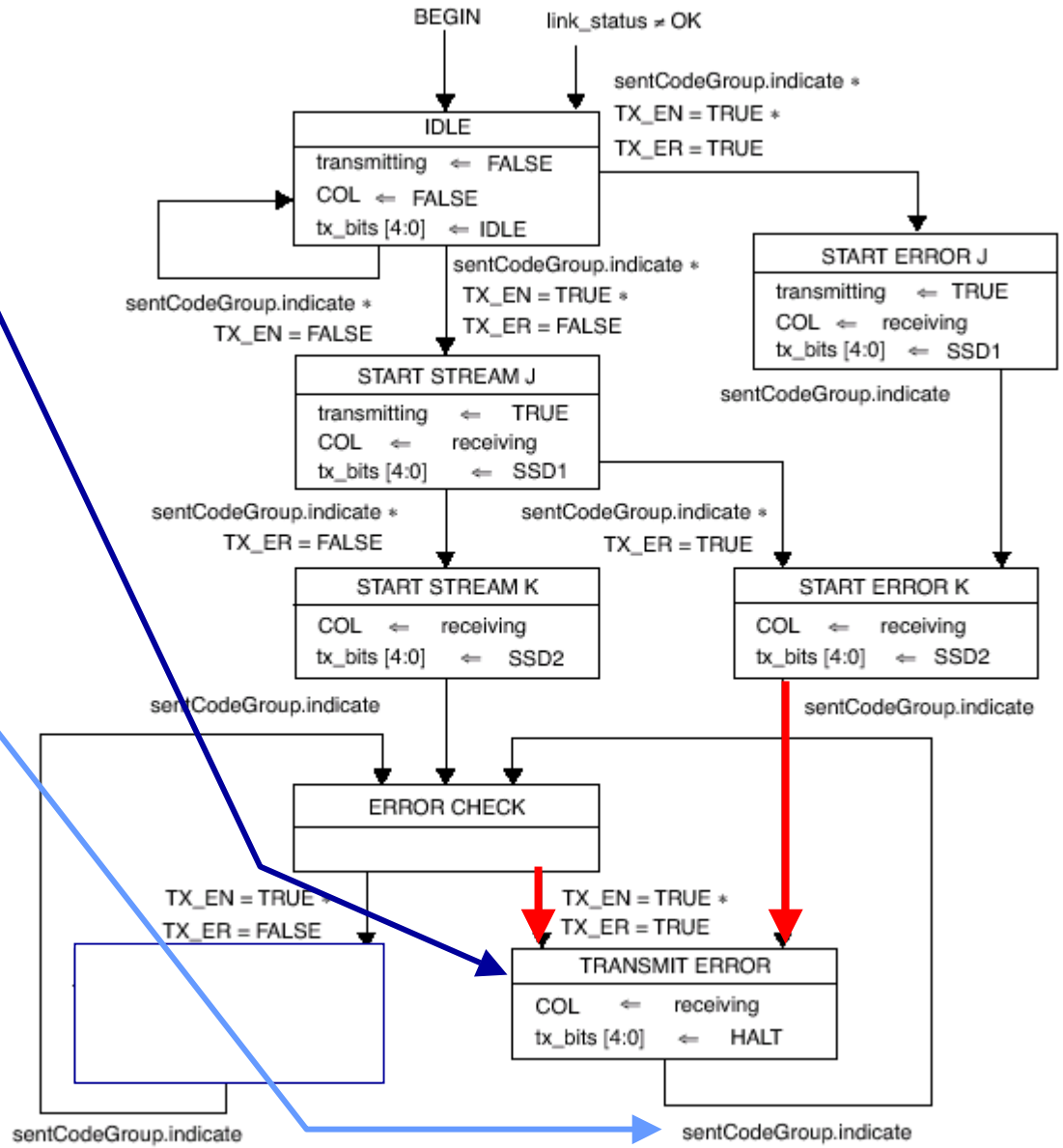
tx_bits[4:0]=HALT

•Transmit /H/ (00100) to force the receiving end to detect an error in the stream.

Exit cases

sentCodeGroup.indicate

•Exit to ERROR CHECK state when another data group is available. If TX_EN and TX_ER are both TRUE then enter the TRANSMIT ERROR state again.



- **Stream Reception**
- Carrier is detected by seeing two non-contiguous ZEROs within any 10-bits of the data stream.
- A valid stream begins with /J/K/.
- A valid stream contains only data code-groups.
- A valid stream ends with /T/R/.

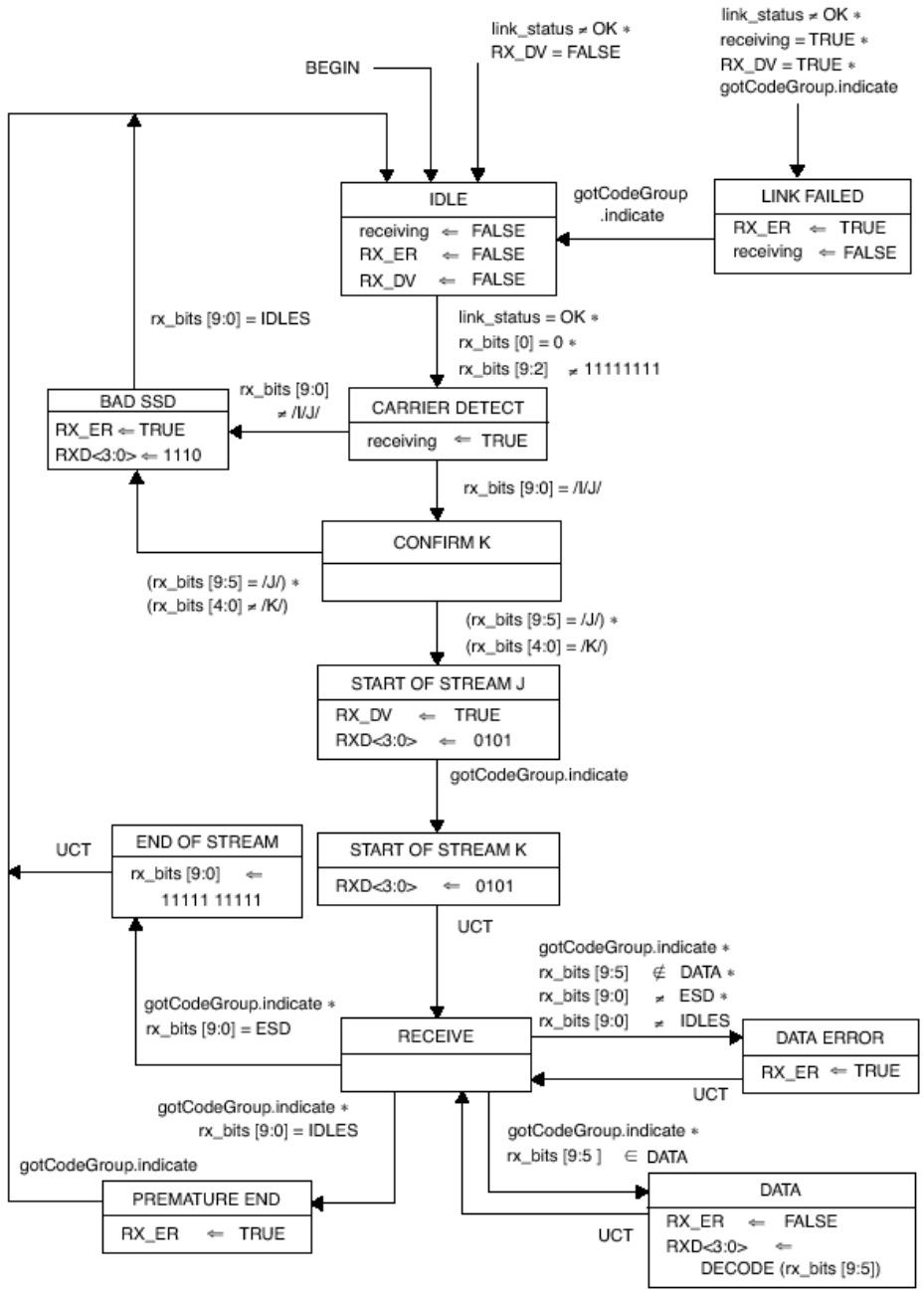


Figure 24–11—Receive state diagram

IDLE

PCS reception initialization state. The PCS also enters this state after a link failure occurs while the PCS is not receiving valid data (link_status≠OK and RX_DV=FALSE).

receiving=FALSE

•PCS is not receiving data

RX_ER=FALSE

•No receive errors

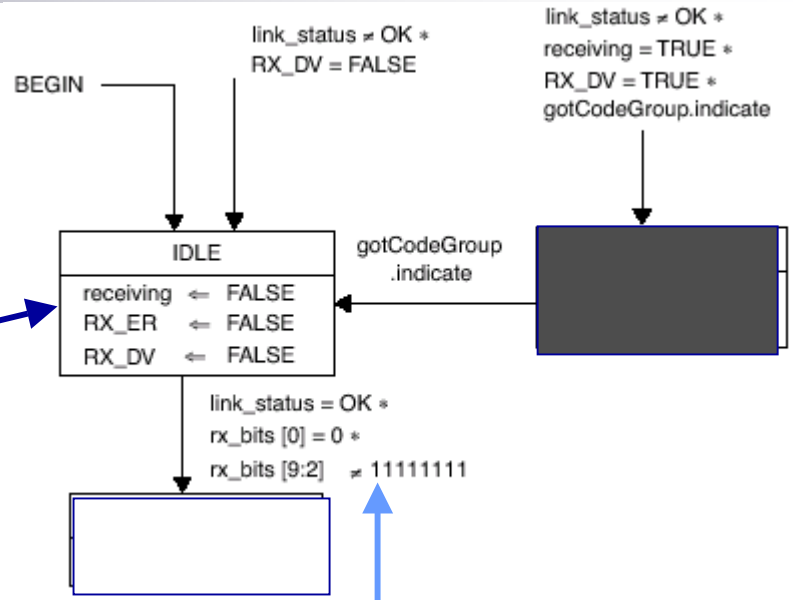
RX_DV=FALSE

•No reception of valid data. The value on the MII data signals are undefined at this point but are ignored by the Reconciliation Sublayer until RX_DV is asserted.

Exit cases

link_status=OK and rx_bits[0]=0 and rx_bits[9:2]≠11111111

•Exit the IDLE state when the link is okay, a 0 is in the first rx_bits vector and another 0 is present in the other bit vectors. The two non-contiguous 0's means carrier is detected. Also, exiting the IDLE state depends on a valid link status.

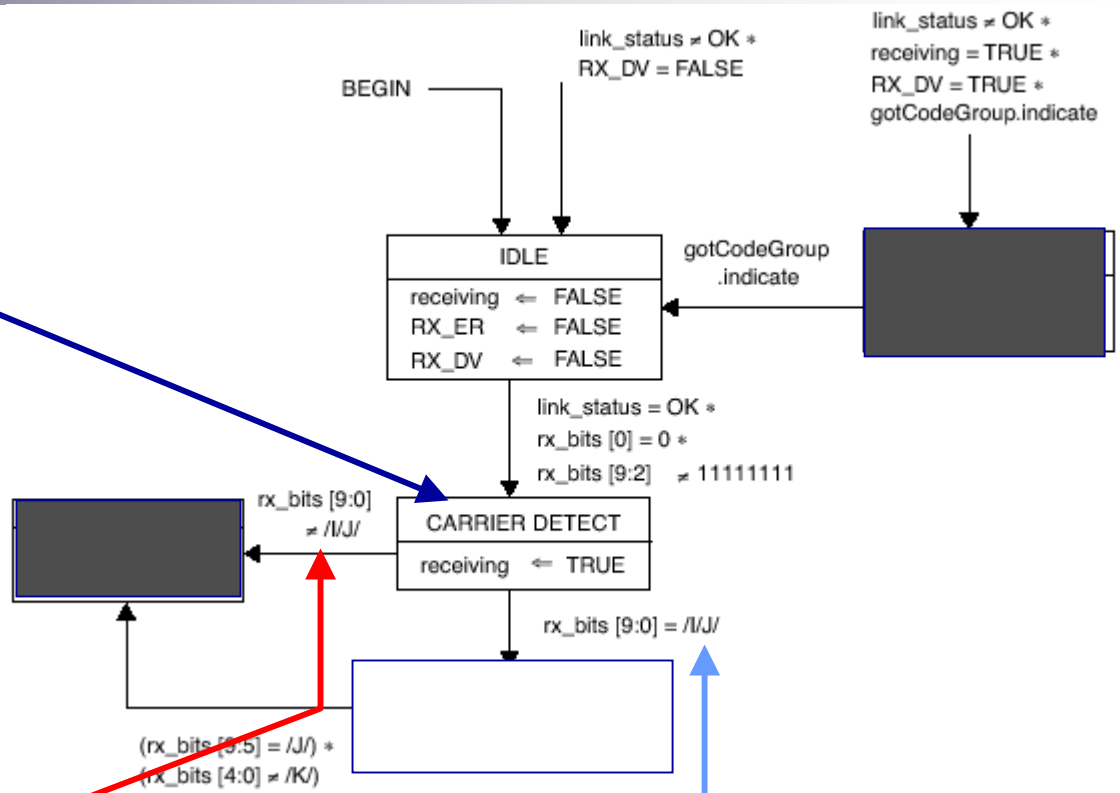


CARRIER DETECT

The PCS detects carrier (something other than idle). The PCS enters this state once it sees two non-contiguous ZEROs within ten bits.

receiving=TRUE

•PCS is now receiving (something)



Exit cases

rx_bits[9:0]≠/I/J/

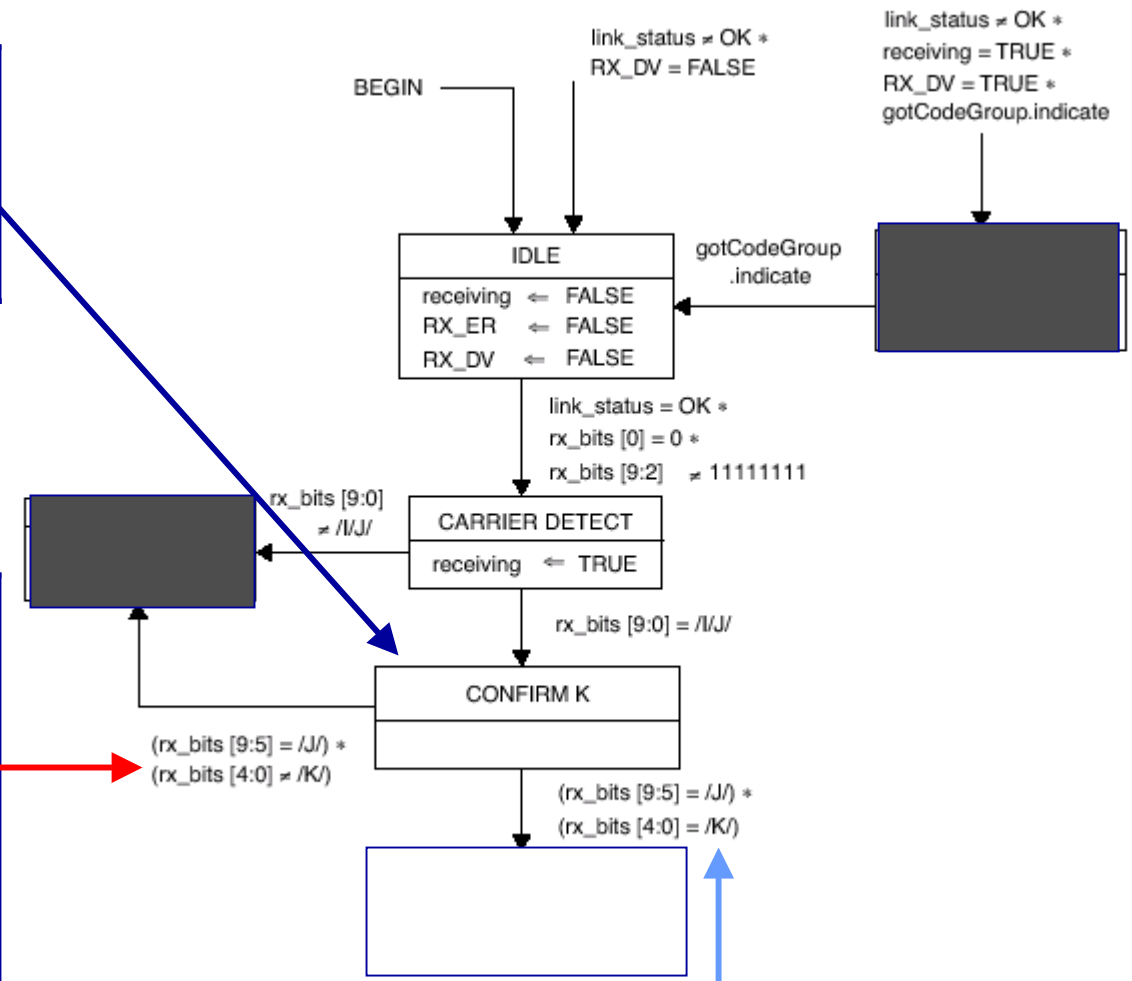
•If the bits in the rx_bits vector do not match the /I/ and /J/ code-groups then move on to the BAD_SSD state. The PCS must detect a complete SSD that indicates a valid stream.

rx_bits[9:0]=/I/J/

•If the bits in the rx_bits vector match the /I/ and /J/ code-groups then move on to the CONFIRM K state. The PCS must detect a complete SSD that indicates a valid stream.

CONFIRM K

Confirm the existence of a valid SSD.



Exit cases

rx_bits[9:5]=/J/ and rx_bits[4:0]≠/K/

•If the bits in the first half of the rx_bits vector match /J/ and the second half does not match /K/ then an invalid SSD has been detected. The PCS moves on to the BAD SSD state.

rx_bits[9:5]=/J/ and rx_bits[4:0]=/K/

•If the bits in the first half of the rx_bits vector match /J/ and the second half matches /K/ then a valid SSD has been detected. The PCS moves on to the START OF STREAM J state.

START OF STREAM J

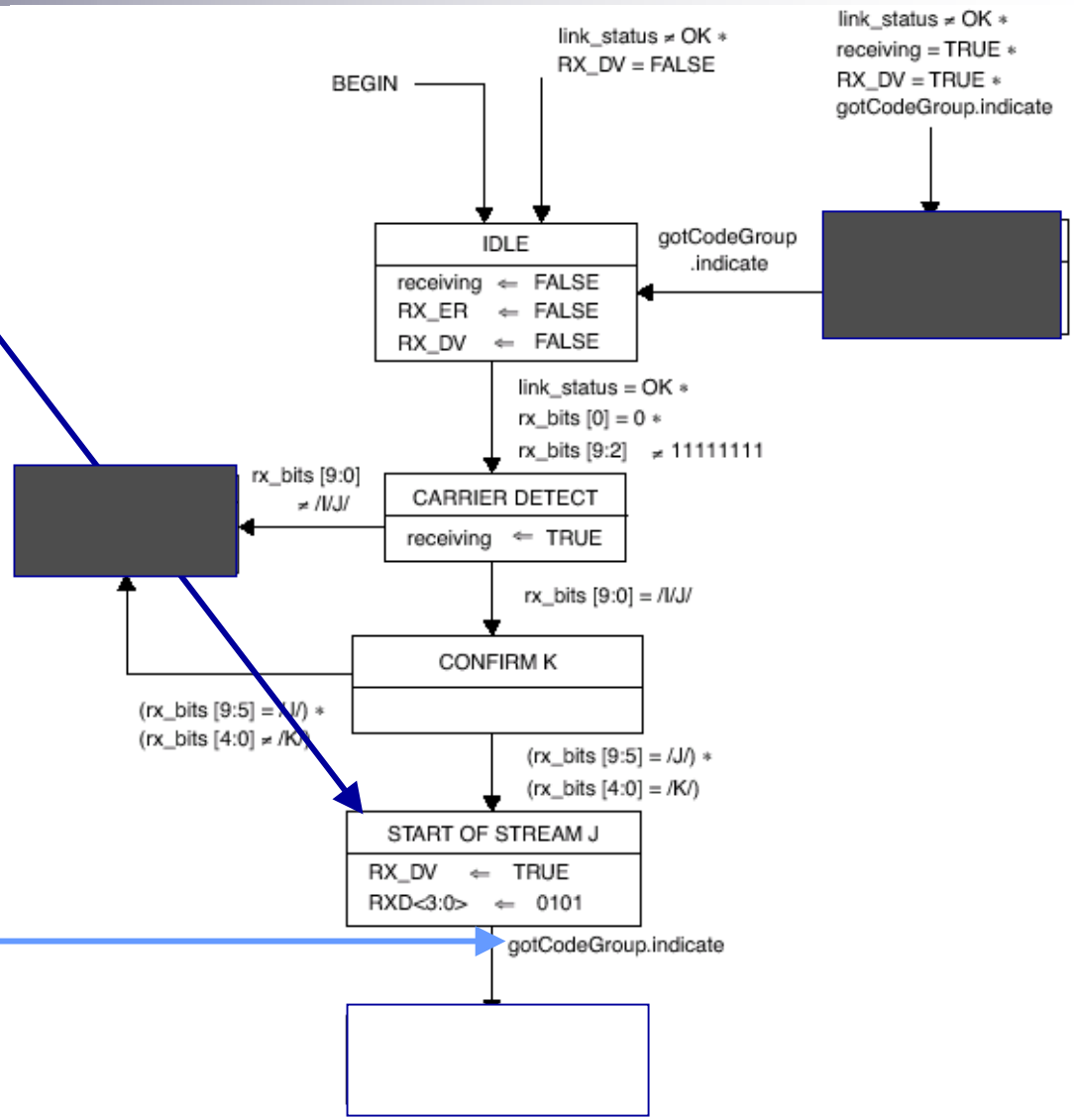
The PCS has detected a valid SSD and will now communicate this to the RS.

RX_DV=TRUE

•PCS asserts RX_DV to indicate the reception of valid data.

RXD<3:0>=0101

•A data 5 is sent in place of the /J/ code-group.



Exit cases

gotCodeGroup.indicate

•After sending the Data 5 the PCS waits for another code-group. This new code-group moves the initial /J/ out of the rx_bits vector.

START OF STREAM K

The PCS has detected a valid SSD and will now communicate this to the RS.

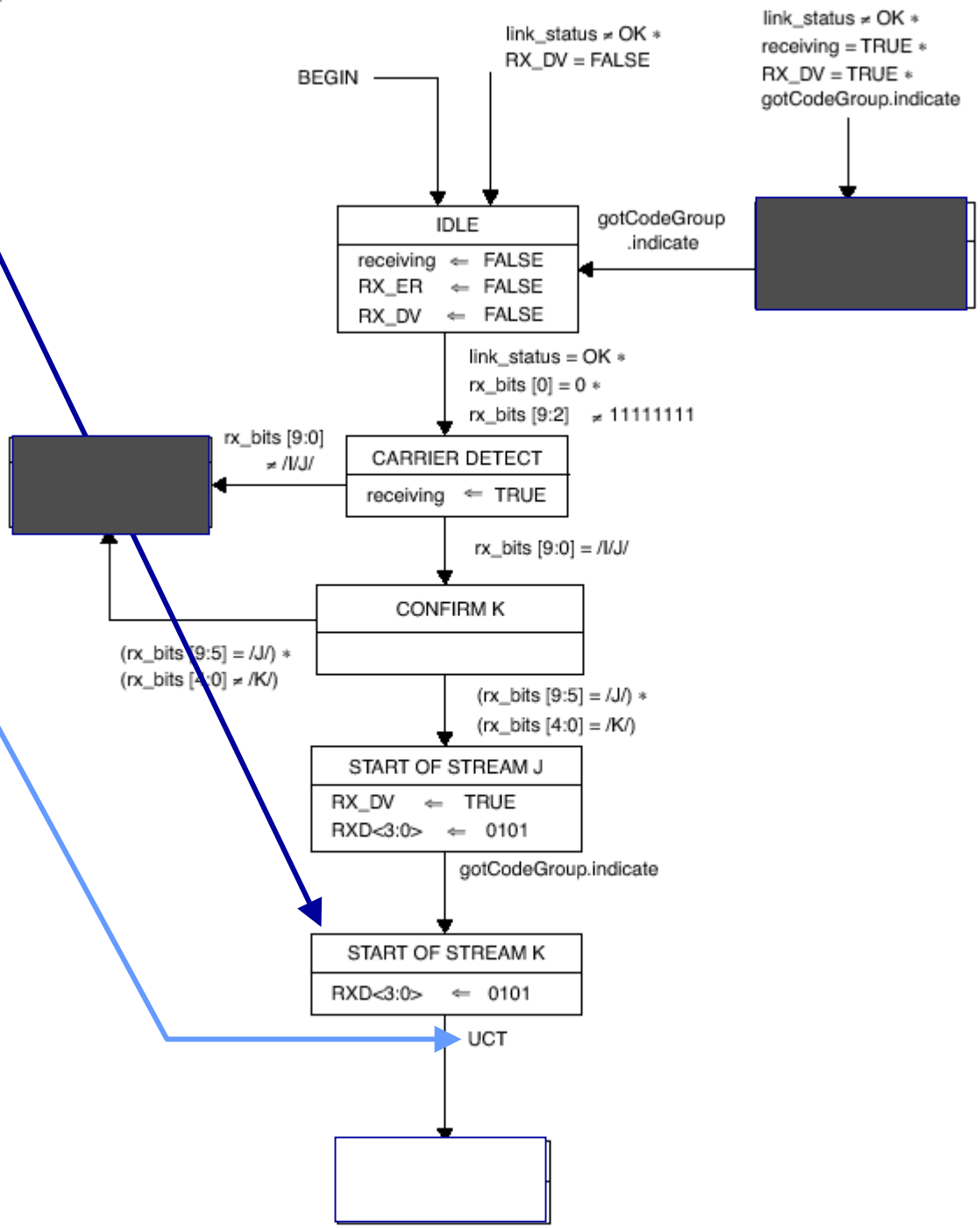
RXD<3:0>=0101

•A data 5 is sent in place of the /K/ code-group.

Exit cases

UCT

•The PCS unconditionally transitions (UCT) to the next state, RECEIVE.



RECEIVE

After receiving a valid SSD the PCS is ready for the stream of data and ESD. The PCS also detects errors in the stream.

Exit cases

gotCodeGroup.indicate and rx_bits[9:0]=ESD

•The PCS receives a valid ESD.

gotCodeGroup.indicate and rx_bits[9:5] DATA and rx_bits[9:0]≠ESD and rx_bits[9:0]≠IDLES

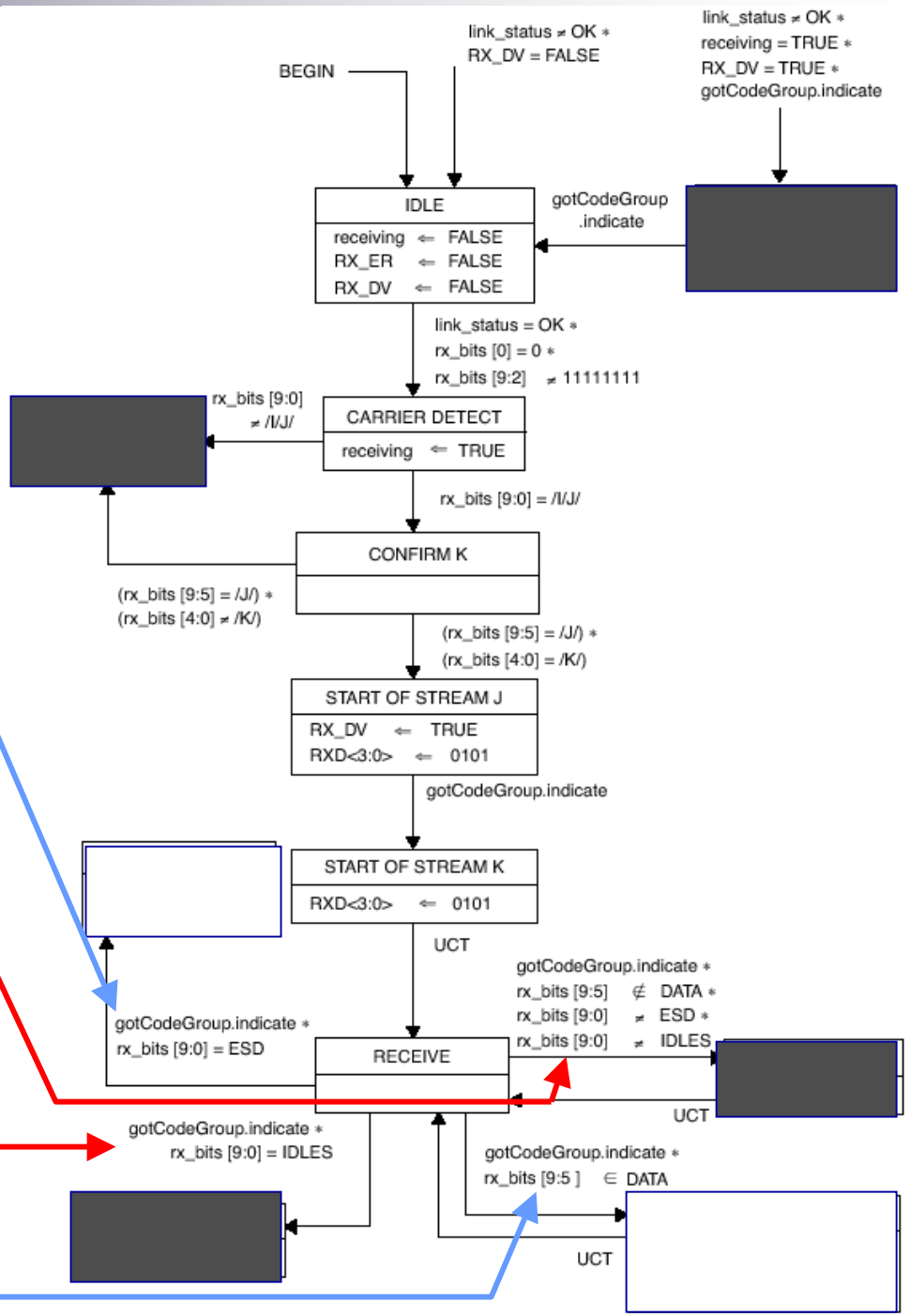
•The PCS receives something other than ESD, IDLE, or data. Enter DATA ERROR state.

gotCodeGroup.indicate and rx_bits[9:0]=IDLES

• The PCS receives idle before an ESD. Enter the PREMATURE END state.

gotCodeGroup.indicate and rx_bits[9:5] ∈ DATA

•The PCS receives a code-group that is a member of the DATA set of code-groups.



DATA

The PCS receives a code-group that is a member of the DATA set of valid code-groups. The PCS decodes it into its 4-bit representation.

RX_ER=FALSE

•Upon reception of a valid data code-group the PCS sets the receive error indication to FALSE.

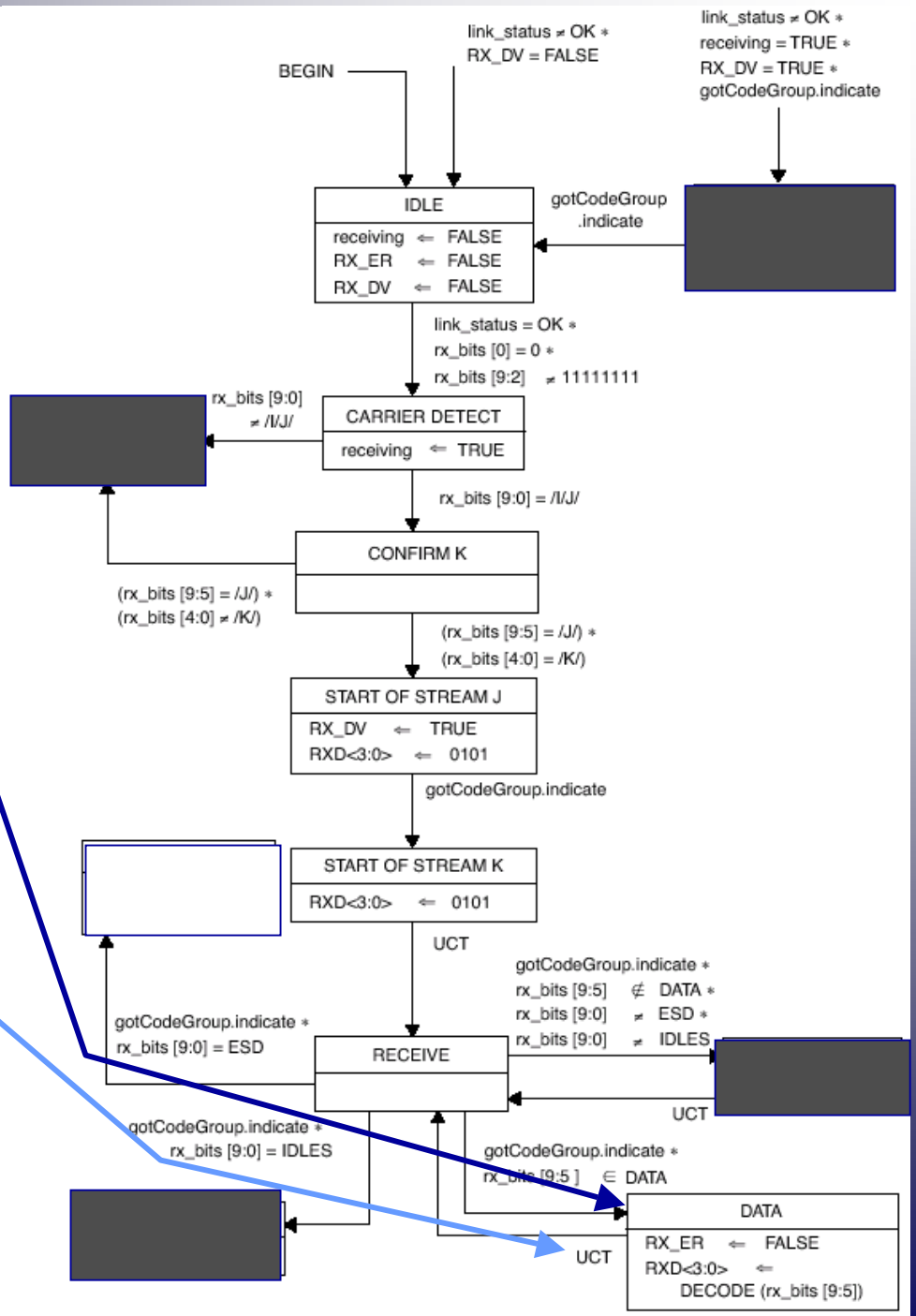
RXD<3:0>=DECODE(rx_bits[9:5])

•The PCS decodes the 5-bit code-group into the 4-bit data representation and sends it to the RS.

Exit cases

UCT

•The PCS unconditionally transitions (UCT) back to the RECEIVE state to receive more data.



END OF STREAM

The PCS has detected a valid ESD and has entered this state.

rx_bits[9:0]=11111 11111

•The PCS waits in this state until it sees two /I/ code-groups. This means the PCS ignores any code-groups that come after the ESD that are not idle.

Exit cases

UCT

•The PCS unconditionally transitions (UCT) back to the beginning IDLE state after seeing idle. It sets RX_DV to FALSE so that the RS knows the stream has ended and it can now ignore the MII data signals.

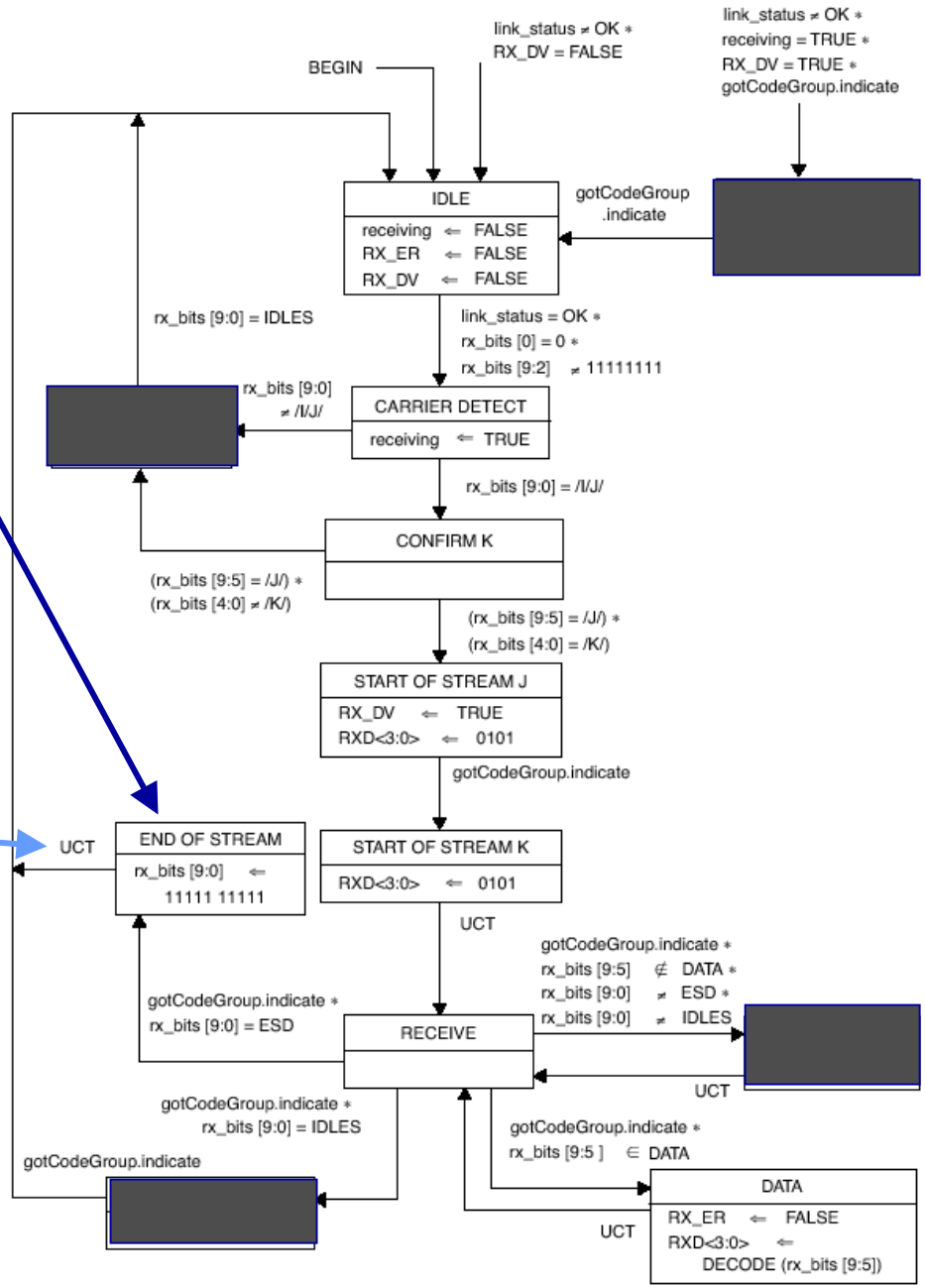


Figure 24–11—Receive state diagram

- **Reception Errors**
- Receive errors can happen during the /J/K/, during the data, or during the /T/R/.
- RX_ER is asserted to signal an error.
- The Reconciliation Sublayer should make sure that when both RX_ER and RX_DV are asserted the MAC will detect an error in the frame.

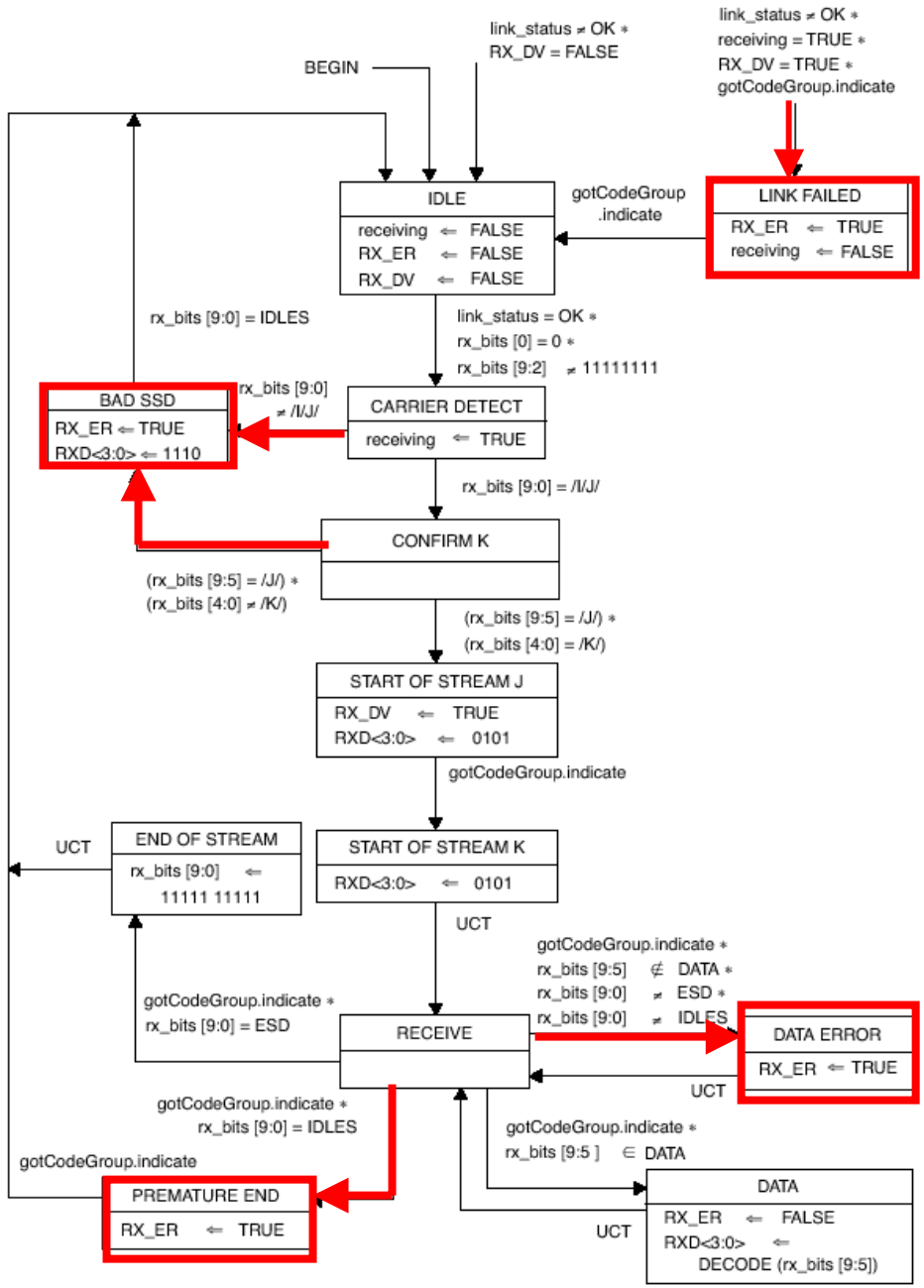


Figure 24–11—Receive state diagram

LINK FAILED

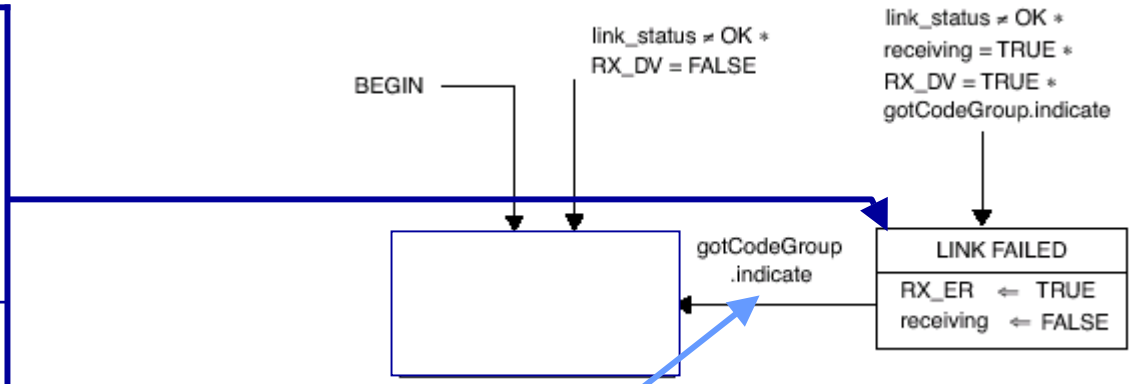
The PCS visits this state when it is receiving valid data and an error in the link occurs (link_status≠OK). This state makes sure an error is indicated before transitioning to the IDLE state.

RX_ER=TRUE

•Since the link has gone bad the PCS signals an error.

receiving=FALSE

•PCS is no longer receiving valid data.



Exit cases

gotCodeGroup.indicate

•When a new code-group is available transition to the IDLE state

CARRIER DETECT Exit cases

rx_bits[9:0]≠/I/J/

•If the bits in the rx_bits vector do not match the /I/ and /J/ code-groups then an invalid SSD has been detected.

BAD SSD

The PCS detects carrier (something other than idle) but a valid SSD does not exist.

RX_ER=TRUE

•The PCS signals an error via RX_ER.

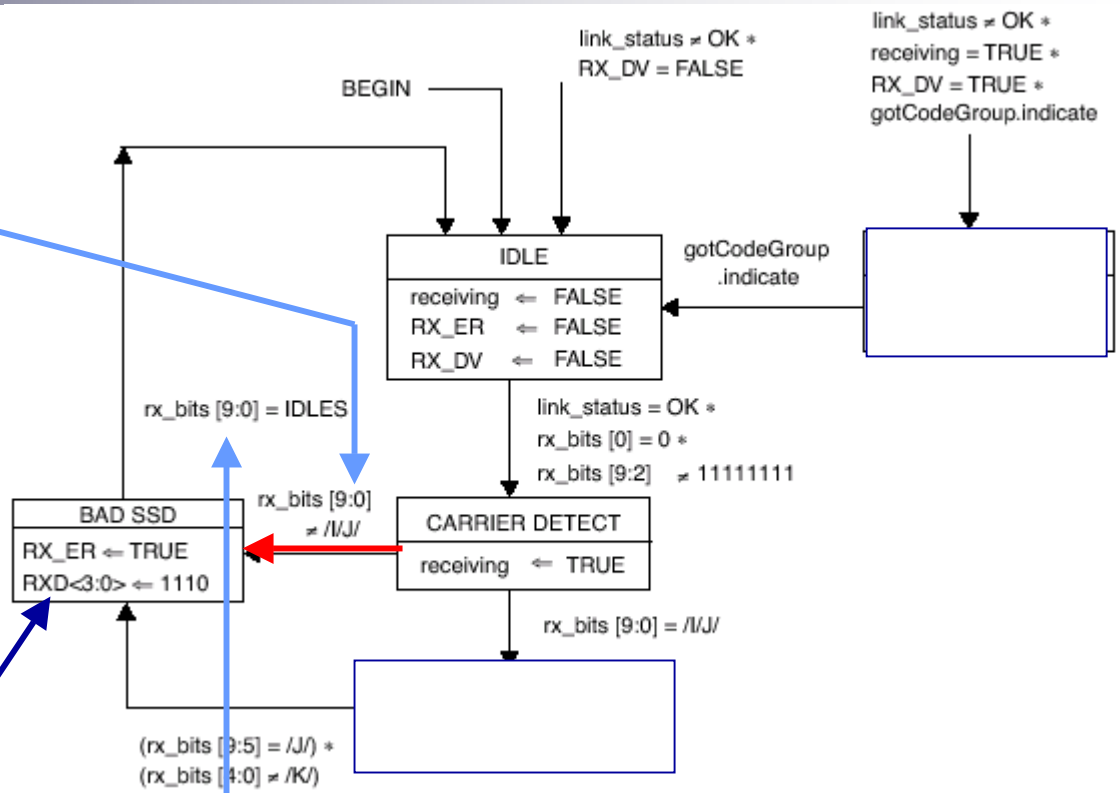
RXD<3:0>=1110

•The PCS sends a data E to higher layers. RX_DV is not asserted in this state, so when RX_ER is asserted with a data E and RX_DV is not asserted a false carrier event has occurred.

Exit cases

rx_bits[9:0]=IDLES

•The PCS continues to send the false carrier indication to higher layers until it sees idle code-groups. It then transitions back to the IDLE state.



CONFIRM K Exit cases

rx_bits[9:5]=/J/ and rx_bits[4:0]≠/K/

•If the bits in the rx_bits vector match /J/ but not /K/ then an invalid SSD has been detected.

BAD SSD

The PCS detects carrier (something other than idle) but a valid SSD does not exist.

RX_ER=TRUE

•The PCS signals an error via RX_ER.

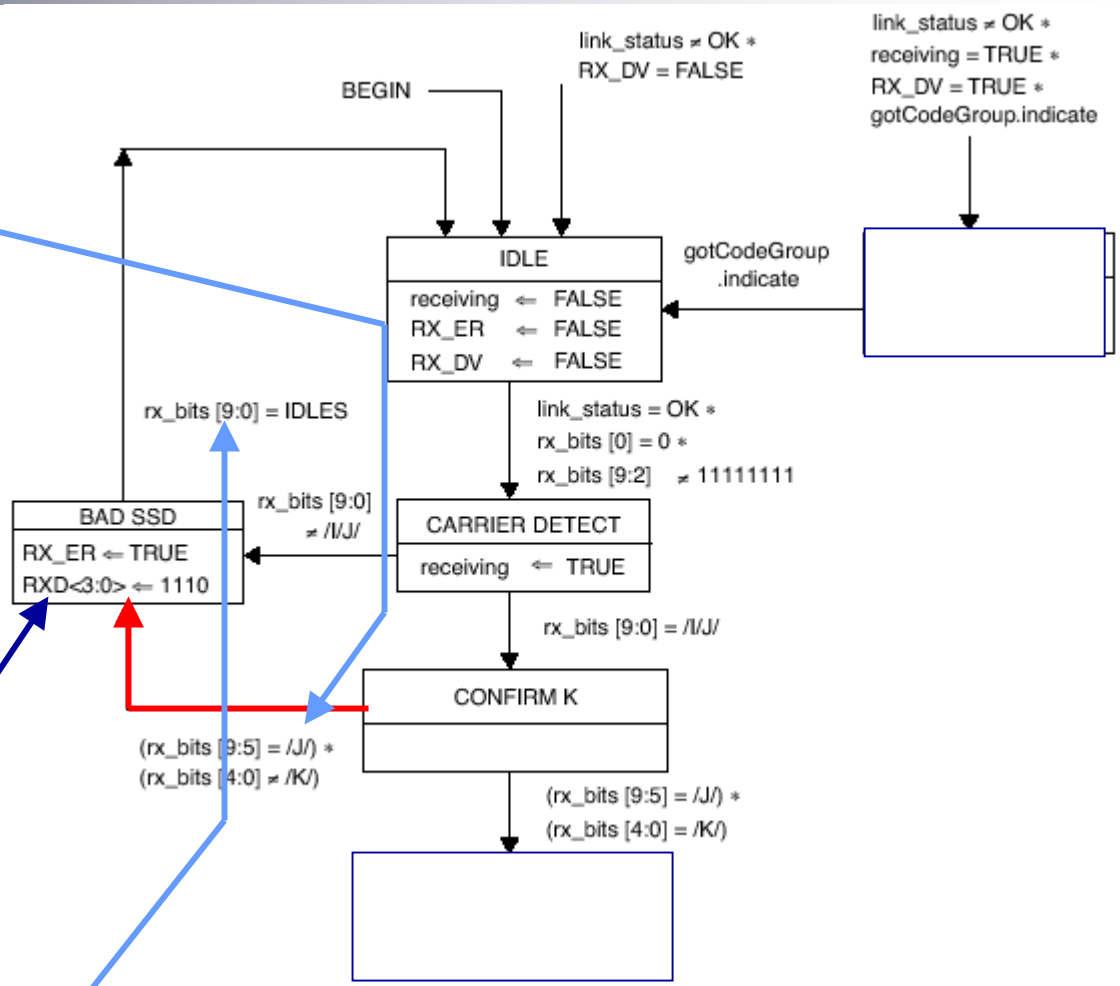
RXD<3:0>=1110

•The PCS sends a data E to higher layers. RX_DV is not asserted in this state, so when RX_ER is asserted with a data E and RX_DV is not asserted a false carrier event has occurred.

Exit cases

rx_bits[9:0]=IDLES

•The PCS continues to send the false carrier indication to higher layers until it sees idle code-groups. It then transitions back to the IDLE state.



RECEIVE Exit cases

gotCodeGroup.indicate and
rx_bits[9:5] DATA and
rx_bits[9:0]≠ESD and
rx_bits[9:0]≠IDLES

•The PCS receives something other than a valid ESD, two idle code-groups, or data. Enter DATA ERROR state.

DATA ERROR

The incoming stream contains an invalid data code-group. An error condition has occurred.

RX_ER=TRUE

•The PCS signals an error via RX_ER. RX_DV is also asserted in this state. Since an invalid code-group has no 4-bit representation the previous value on RXD<3:0> should still be asserted along with RX_ER and RX_DV.

Exit cases

UCT

•The PCS unconditionally transitions (UCT) back to the RECEIVE state to receive more data.

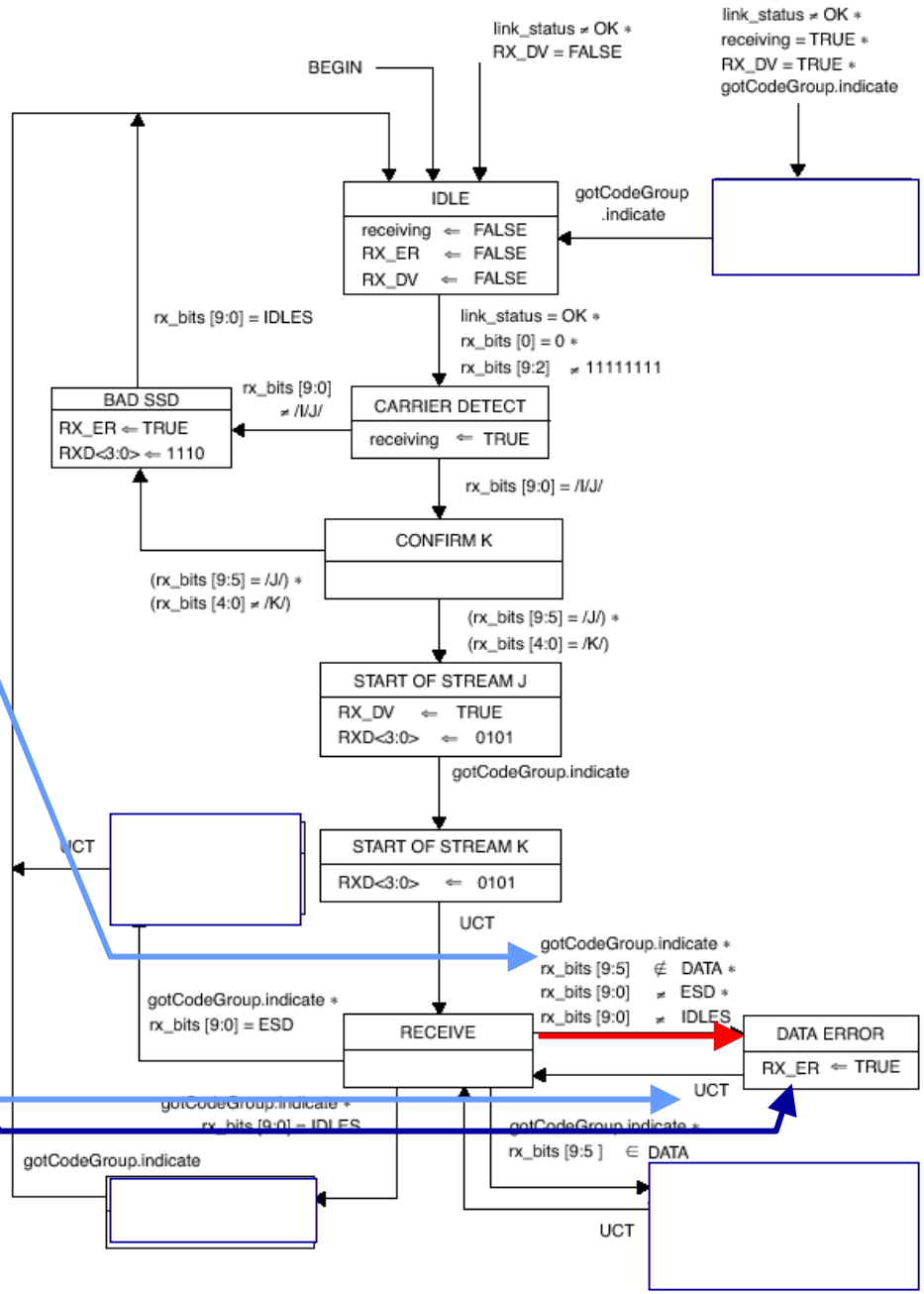


Figure 24–11—Receive state diagram

RECEIVE Exit cases

gotCodeGroup.indicate and rx_bits[9:0]=IDLES

•The PCS transitions out of the RECEIVE state when it sees two idle code-groups (/I/I/). The PCS expects to see valid data or a valid ESD while in the RECEIVE state.

PREMATURE END

The incoming stream has transitioned from data code-groups to idle code-groups without a valid End of Stream Delimiter. An error condition has occurred.

RX_ER=TRUE

•The PCS signals an error via RX_ER. RX_DV is also asserted in this state.

Exit cases

gotCodeGroup.indicate

•When another code-group is available the PCS transitions back to the IDLE state.

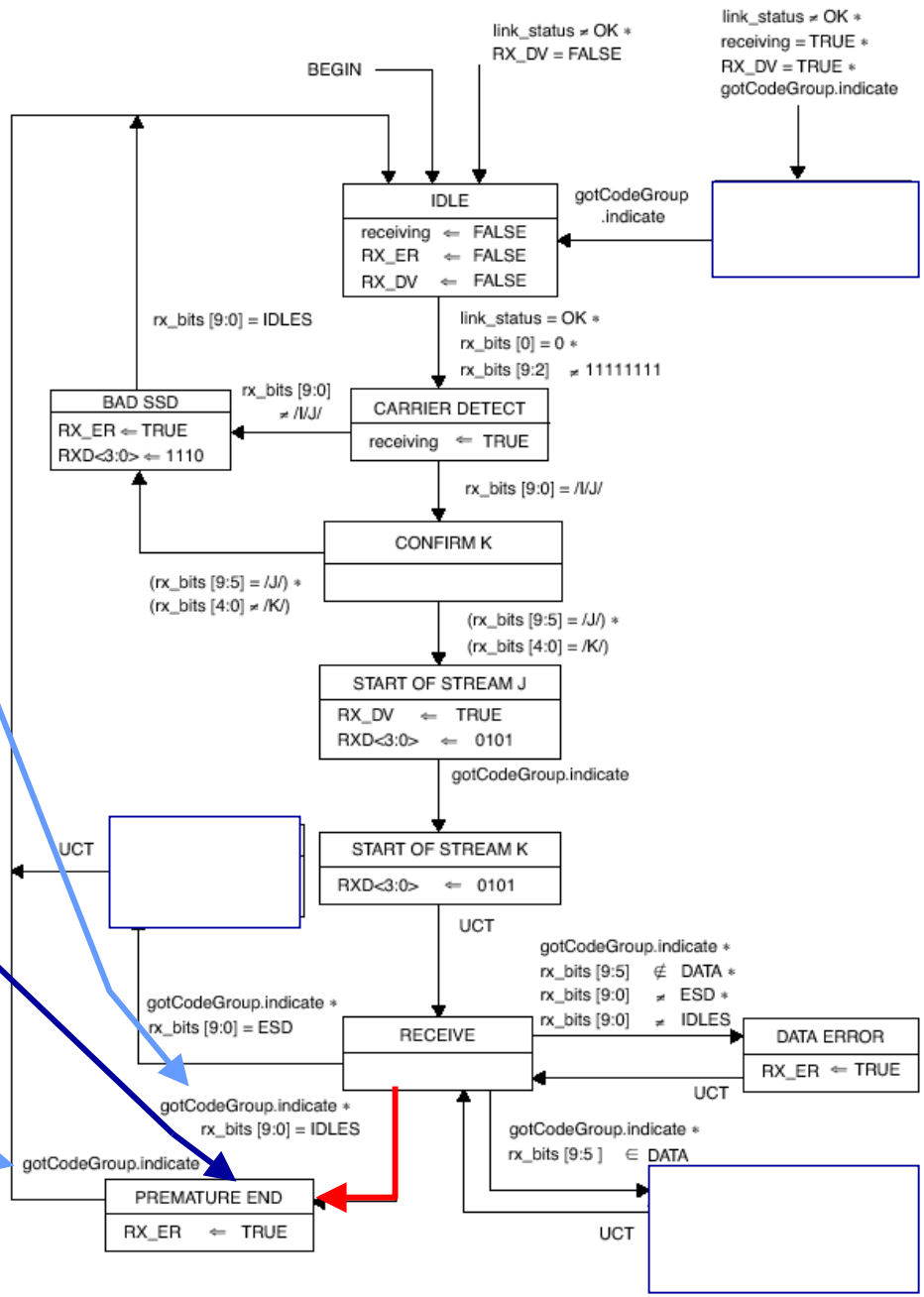


Figure 24–11—Receive state diagram

In Summary

- PCS performs 4B/5B encoding and decoding
- PCS signals carrier sense and collisions
- Stream serialization to the underlying PMA
- Mapping of the MII signals to the PMA
- Detection of errors in the incoming data stream

Additional resources

- ...

References

- For correct citation format for virtually any source see: http://www.reference.unh.edu/bib.html