10 Mb/s Single Twisted Pair Ethernet

10BASE-T1L Draft Overview

Steffen Graber
Pepperl+Fuchs

Content

- PCS
  - Overview
  - Ternary Symbol Coding
  - Running Disparity Handling and Checking
  - Telegram Synchronization
  - Side Stream Scrambling
  - Signalizing during Idle Transmission
  - Symbol Boundary Detection
  - Polarity Detection
  - Descrambler Synchronization
  - Detection of Data and Idle Transmission
  - Transmit Symbol Generation Block Diagram
  - Transmit State Machine
  - Receiver Symbol Decoding Block Diagram
  - Receive State Machine
  - Jabber State Machine
  - MII Interface
- PMA
  - Overview
  - PHY Control State Machine
  - Link Monitor State Machine
  - Transmitter Test Modes
  - Transmitter Test Fixture
  - Transmitter Electrical Specifications
  - Receiver Electrical Specifications
  - Delay Constraints
- Management Interface
- Informative Annex for Intrinsic Safety

PCS Overview

- Ternary (PAM-3) Symbol Coding (+1, 0, -1).
- Symbol Rate of 7.5 MSPS/s using 4B3T Coding.
- Running Disparity Handling and Checking.
- Telegram Synchronization using robust Comma Sequences.
- Side Stream Scrambling.
- Automatic Descrambler Synchronization.
- Detection of Data and Idle Transmission.
- Signalizing of „Remote Receiver Status“ and „Low Power Idle Request“ included in Idle Data.
- MII Host Interface.

- Currently Energy Efficient Ethernet is not implemented (will need further modifications to state machines).

Ternary Symbol Coding

- PAM-3 Symbol Coding with 7.5 MSPS/s and controlled running disparity.
- 4 Bit (MII Nibble Data) are coded in a triple ternary symbol.
- Depending on the actual disparity (accumulated difference to mean value of the signal) the right coding word is being used.

$Sd_n[3:0]$	Disparity = 1	Disparity = 2	Disparity = 3	Disparity = 4
0000	+0+	0-0	0-0	0-0
0001	0-+	0-+	0-+	0-+
0010	+-0	+-0	+-0	+-0
0011	00+	00+	00+	--0
0100	-+0	-+0	-+0	-+0
0101	0++	-00	-00	-00
0110	-++	-++	--+	--+
0111	-0+	-0+	-0+	-0+
1000	+00	+00	+00	0--
1001	+-+	+-+	+-+	---
1010	++-	++-	+--	+--
1011	+0-	+0-	+0-	+0-
1100	+++	-+-	-+-	-+-
1101	0+0	0+0	0+0	-0-
1110	0+-	0+-	0+-	0+-
1111	++0	00-	00-	00-

Running Disparity Handling and Checking

- To limit the maximum additional signal amplitude on the link segment, caused by BLW, a strict control of the running disparity is implemented by the used 4B3T coding.
- Depending on the current running disparity the transmitted code words are always chosen in a way, so that the running disparity is limited to a narrow band between 1 and 4.

- Doing a reset of the running disparity within the comma words and calculating the running disparity also within the receiver allows an easy additional error detection possibility at the receiver side.
- Even a single bit error leads to a difference in the calculated running disparity at the receiver side, and it is likely, that the subsequently received bit patterns do not fit the calculated running disparity at the receiver side anymore and therefore are already detected as a receive error before feeding the data to the MII interface.

Telegram Synchronization

- Telegram Synchronization is done using comma sequences.
- Each comma sequence consists of four triple ternary symbols (12 PAM-3 Symbols, 16 Bits)

- COMMA1 and COMMA2 use the triple ternary symbol (0, 0, 0), which is not being used within the normal data stream (during normal communication a maximum of 4 subsequent zeros is seen in the data stream).
- As the triple ternary symbol boundary is adjusted during the training process, theoretically just COMMA1 or COMMA2 would be enough for signalizing, adding a second comma word increases the robustness of the delimiter for false detection.
- The next triple ternary symbol is DISPRESET3, which is chosen depending on the current running disparity.
- A total of 4 different DISPRESET3 symbols exist, as there are four possible running disparity values.
- The fourth triple ternary symbol provides information, if the delimiter is an SSD, an ESD or an ESD with Error.

Telegram Synchronization

- The following table shows the different possible DISPRESET3 symbols:

Symbol Triplet	Disparity = 1	Disparity = 2	Disparity = 3	Disparity = 4
DISPRESET3	(-1, 0, 1)	(-1, 0, 0)	(-1, 0, -1)	(-1, -1, -1)

- The symbols are chosen so that, depending on the current running disparity, a running disparity value of 1 is always reached (together with the SSD4/ESD4/ESD_ERR4 symbols this will lead to a running disparity of 2, as these symbols always have a disparity of 1).

- The following table shows the different possible SSD4/ESD4/ESD_ERR4 symbols:

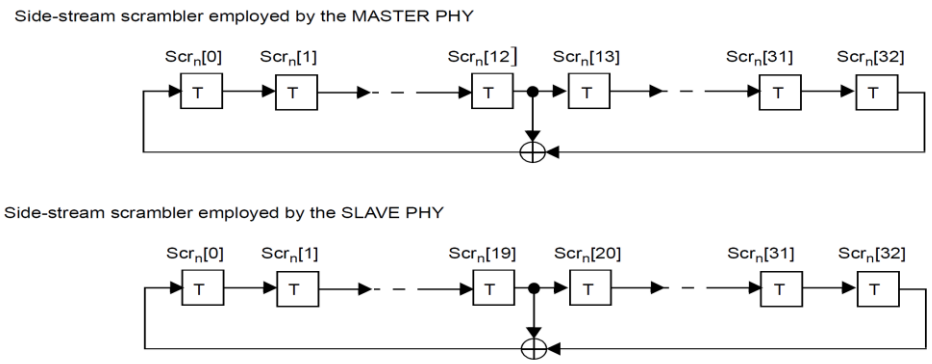
Delimiter	(TA_n, TB_n, TC_n)
SSD4	(1, 1, -1)
ESD4	(1, -1, 1)
ESD_ERR4	(-1, 1, 1)

- The coding of the ternary delimiter symbols is chosen, so that they have the maximum possible distance within the coding, to detect a wrong delimiter, at least two single symbols have to be flipped.

Side Stream Scrambling

- Parts of the 1000BASE-T (Clause 40) side stream scrambling are being used within the PCS.
- The following generator polynomials are being used within the master and the slave PHY transmitter:

$$g_M(x) = 1 + x^{13} + x^{33}$$
$$g_S(x) = 1 + x^{20} + x^{33}$$



- At the descrambler side master and slave polynomials are exchanged compared to the scrambler side:

$$g'_M(x) = 1 + x^{20} + x^{33}$$
$$g'_S(x) = 1 + x^{13} + x^{33}$$

Side Stream Scrambling

- Additionally to the scrambler polynomials an auxiliary polynomial is being used:

$$g(x) = x^3 \hat{} x^8$$

- With the help of this auxiliary polynomial 4 bits $Sy_n[3:0]$ are being generated:

$$Sy_n[0] = Scr_n[0]$$
$$Sy_n[1] = g(Scr_n[0]) = Scr_n[3] \hat{} Scr_n[8]$$
$$Sy_n[2] = g^2(Scr_n[0]) = Scr_n[6] \hat{} Scr_n[16]$$
$$Sy_n[3] = g^3(Scr_n[0]) = Scr_n[9] \hat{} Scr_n[14] \hat{} Scr_n[19] \hat{} Scr_n[24]$$

- By construction, the four bits $Sy_n[3:0]$ are derived from elements of the same maximum-length shift register sequence of length 2^{33-1} as $Scr_n[0]$, but shifted in time by varying delays. The associated delays are all large and different so that there is no apparent correlation among the bits.

Side Stream Scrambling

- From bits Sy_n[3:0] another four bits Sc_n[3:0] are generated (just to stay within the naming conventions):

$$Sc_n[3:0] = Sy_n[3:0]$$

- From scrambler bits Sc_n[3:0], TXD[3:0] and the control bits, bits Sd_n[3:0] are generated:

$$Sd_n[3] = \begin{cases} Sc_n[3]\text{^}TXD_n[3] & if\ (tx\_enable\_mii = TRUE) \\ Sc_n[3]\text{^}1 & else\ if\ (loc\_rcvr\_status = OK) \\ Sc_n[3] & else \end{cases}$$

$$Sd_n[2] = \begin{cases} Sc_n[2]\text{^}TXD_n[2] & if\ (tx\_enable\_mii = TRUE) \\ Sc_n[1]\text{^}1 & else\ if\ (loc\_lpi\_req = TRUE) \\ Sc_n[1] & else \end{cases}$$

$$Sd_n[1:0] = \begin{cases} Sc_n[1:0]\text{^}TXD_n[1:0] & if\ (tx\_enable\_mii = TRUE) \\ (Sc_n[2]\ Sc_n[0]) & else \end{cases}$$

Bits Sd_n[3:0] are then 4B3T encoded and provided to the PMA.

Signalizing during Idle Transmission

- During transmission of idle data the two upper bits of each 4 bit nibble are used to transmit the local receiver status and an optional local low power idle request signal, needed for a later EEE implementation.

- To provide a reliable data transmission, it is checked that the same information is received at least 8 times in a row during idle transmission with the same value, before the status bit is being changed at the receiver side.

- During idle transmission $Sc_n[1]$ and $Sc_n[2]$ are exchanged compared to a normal data transmission.
- This is necessary to be able to reliably distinguish a „zero“ data transmission from an idle transmission.

Symbol Boundary Detection

- During initial training phase the correct boundary of the triple ternary symbols needs to be detected.
- This can be done independently from the polarity detection and descrambler synchronization.
- Initially the PHYs are only sending idle data without comma sequences, therefore the symbol (0, 0, 0) will not be present within the received data stream.
- If symbol (0, 0, 0) is being received, the receiver knows, that it is not being synchronized to the correct symbol boundary and therefore the input to the symbol decoder needs to be shifted by one position.
- If then a (0, 0, 0) symbol is still being received, another shift will be necessary.
- If there is a receive error and the shift has been performed accidently, another two shift operations will be necessary (the same shifting algorithm can be used).
- Shifting can be done independently from other operations until a descrambler lock is being reached.

Polarity Detection

- Additionally to the symbol boundary detection a signal polarity detection is necessary on the master and the slave PHY side.
- Initially normal polarity can be assumed and when a descrambler lock is not reached within a certain amount of time (e.g. after receiving 128 triple ternary symbols), the polarity at the receiver side is reversed.
- The toggling of the receive polarity is done until a descrambler lock is being reached.
- Polarity detection can be done in parallel to the symbol boundary detection.
- If there is a receive bit error during polarity detection, the scrambler lock will not be reached and another polarity toggle cycle is necessary.
- The polarity of the transmitted signal is not being changed.
- Therefore the slave as well as the master PHY have to implement the same polarity detection algorithm.
- This allows for an independent polarity detection on the master as well as on the slave side.

Descrambler Synchronization

- During idle transmission bits $Sd_n[1:0]$ contain only the raw scrambler data $Sc_n[2]$ and $Sc_n[0]$ (only $Sd_n[3:2]$ contains control information XORed with the scrambler data).
- Therefore the transmitted $Sd_n[0]$ is identical to $Sc_n[0]$ of the side stream scrambler at the transmitter side.
- During descrambler synchronization, this value can be used to load the side stream scrambler at the receiver side with the correct data.
- As the side stream scrambler has a length of 33 bit, after 33 triple ternary symbols have been received (and no bit error is assumed) the side stream scrambler at the receiver side is in sync with the side stream scrambler at the transmitter side.
- To check, that there has been no error during synchronization, bits $Sd_n[1:0]$ can be checked against the descrambler output for some time (e.g. at least for 33 triple ternary symbols, as the descrambler has a length of 33 bit), to make sure that the descrambler is locked correctly.

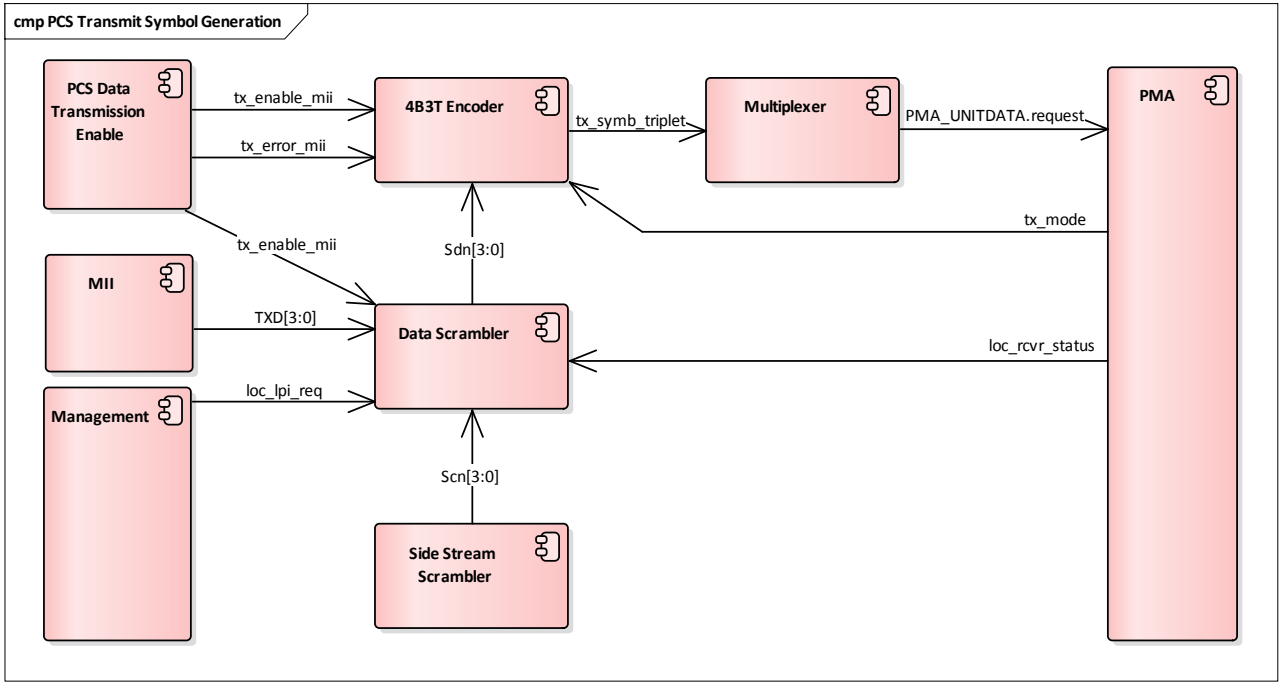
Detection of Data and Idle Transmission

- During normal communication a data transmission starts after receiving a SSD (start of stream delimiter), idle data transmission starts after receiving an ESD (end of stream delimiter).
- All comma sequences contain a high number of bits and are checked to be valid over the full length of the sequence for bit errors, therefore it is very unlikely to accidently detect a delimiter within a normal data or idle stream.
- Nevertheless it could happen that during transmission of a delimiter a bit error occurs.
- If such a case is detected, the receive state machine waits until idle data are detected again.

- As within the idle data stream the lower two bits of a 4 bit nibble always contain the content of scrambler bits 2 and 0, after descrambling these bits always must be zero, if idle data are being transmitted.
- Analyzing, that e.g. at least the last 8 received and decoded triple ternary symbols in a sequence fulfil this requirement, idle data transmission is assumed and the receiver is set to IDLE state again.

Transmit Symbol Generation Block Diagram

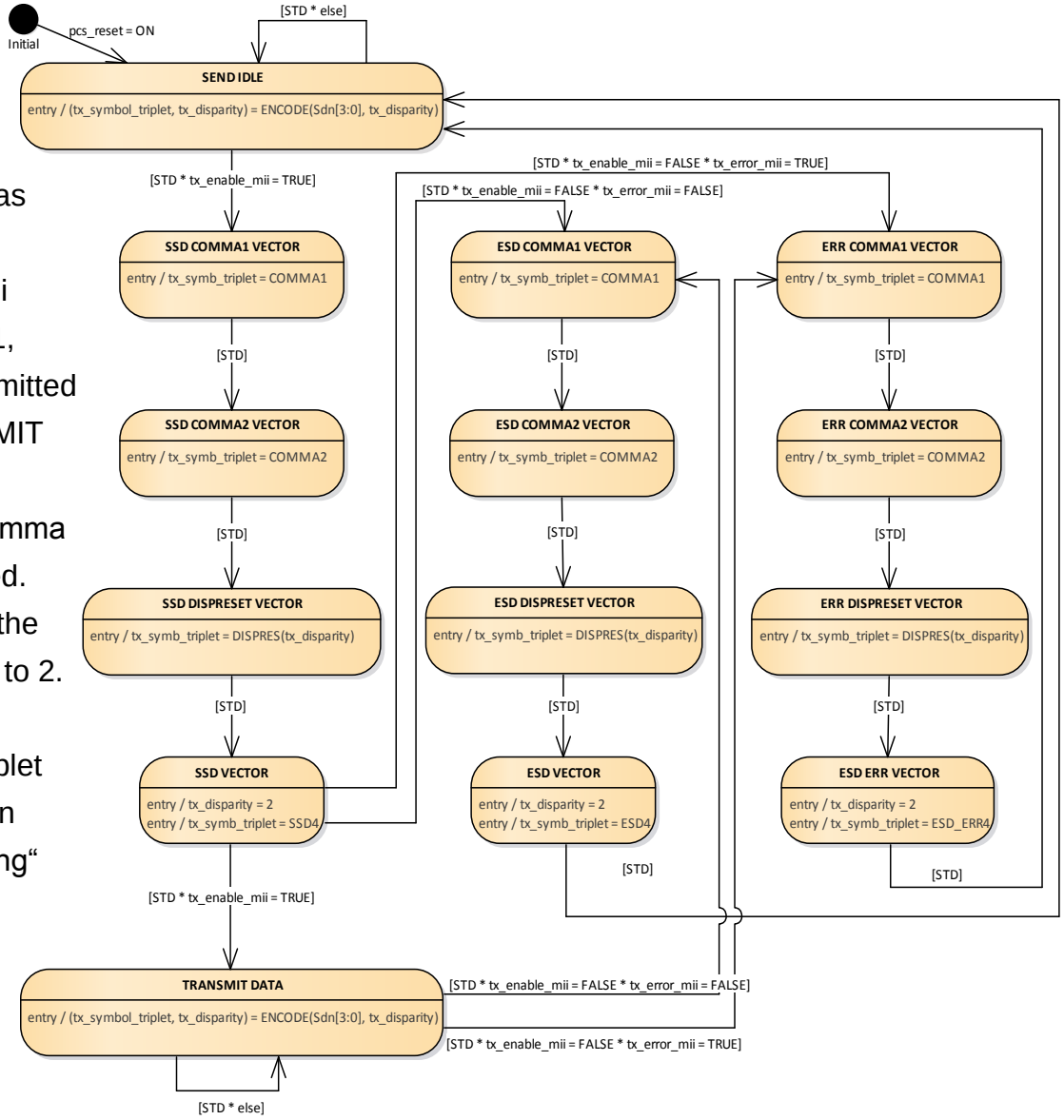
- The following block diagram shows an overview of the generation of the transmit symbols, which is done within the PCS:



- The transmit state machine on the next slide is used to control the different function blocks, depending on the control signals coming from the MII interface, the PMA layer or being created within the PCS layer.

Transmit State Machine

- There are two main states „SEND IDLE“ and “TRANSMIT DATA”.
- „STD“ is an alias for symbol_triplet_timer_done and is triggered each time a tx_symbol_triplet has been transmitted.
- Being in „SEND IDLE“ as soon as tx_enable_mii becomes TRUE, a comma sequence (COMMA1, COMMA2, DISPRESET3, SSD4) is being transmitted and afterwards the state machine is in „TRANSMIT DATA“ state.
- If tx_enable_mii becomes „FALSE“ a second comma sequence (with ESD4/ESD_ERR4) is transmitted.
- While transmitting SSD4, ESD4 or ESD_ERR4 the running disparity on the transmitter side is reset to 2.
- During transmission of idle or data symbols, the ENCODE function, calculates the tx_symbol_triplet from the binary 4 bit values of $Sd_n[3:0]$, based on the table shown in chapter „ternary symbol coding“ within this presentation.
- Additionally this function also calculates the tx_disparity, based on the disparity (DC offset) of each individual tx_symbol_triplet.



Transmit State Machine

- Within the transmit state machine the ENCODE function is being used:

$$(tx\_symbol\_triplet, tx\_disparity) = ENCODE(Sd_n[3:0], tx\_disparity)$$

- This function encodes the nibble data $Sd_n[3:0]$ provided by the scrambler, based on the current tx_disparity (transmit running disparity).
- The tx_disparity can be between 1 and 4 and the respective tx_symbol_triplet is taken from the 4B3T encoding table shown within this presentation in chapter „Ternary Symbol Coding“ based on the $Sd_n[3:0]$ value and the tx_disparity:

$$tx\_symbol\_triplet = table_{4B3T}(Sd_n[3:0], tx\_disparity)$$

- The second output value of this function is an updated tx_disparity value.
- This value is calculated in the following way:

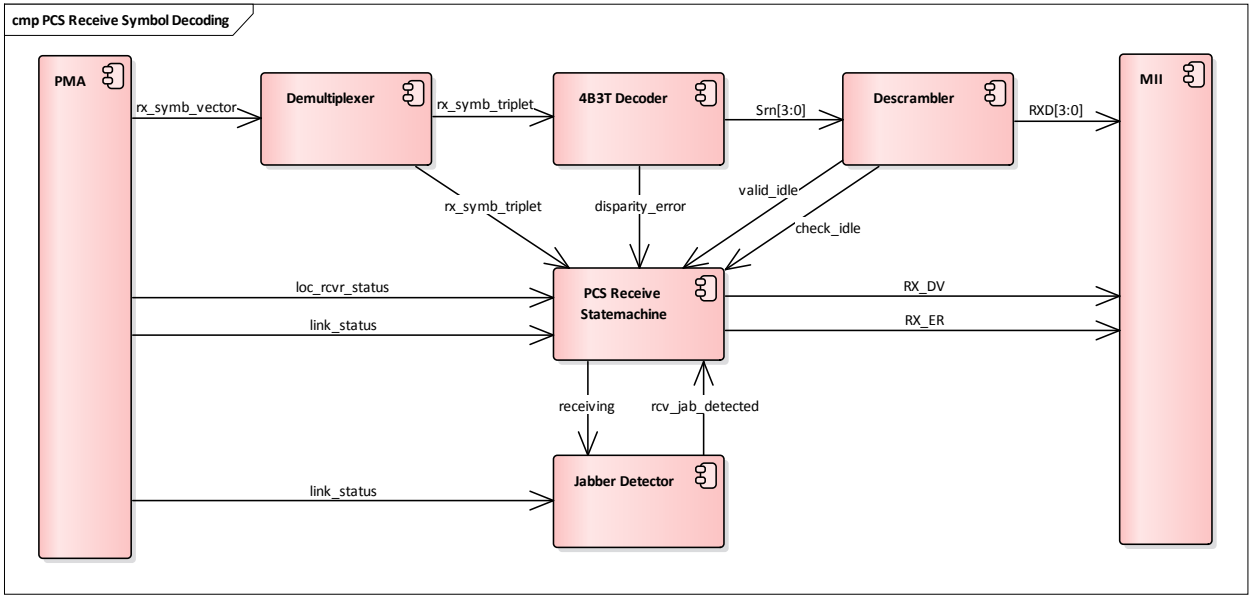
$$tx\_disparity = tx\_disparity + disparity\ of\ currently\ encoded\ tx\_symbol\_triplet$$

- The function DISPRES returns one of the four possible DISPRESET3 triple ternary symbols (see chapter „Ternary Symbol Coding“ within this presentation), depending on the actual tx_disparity:

$$tx\_symbol\_triplet = table_{DISPRESET3}(tx\_disparity)$$

Receiver Symbol Decoding Block Diagram

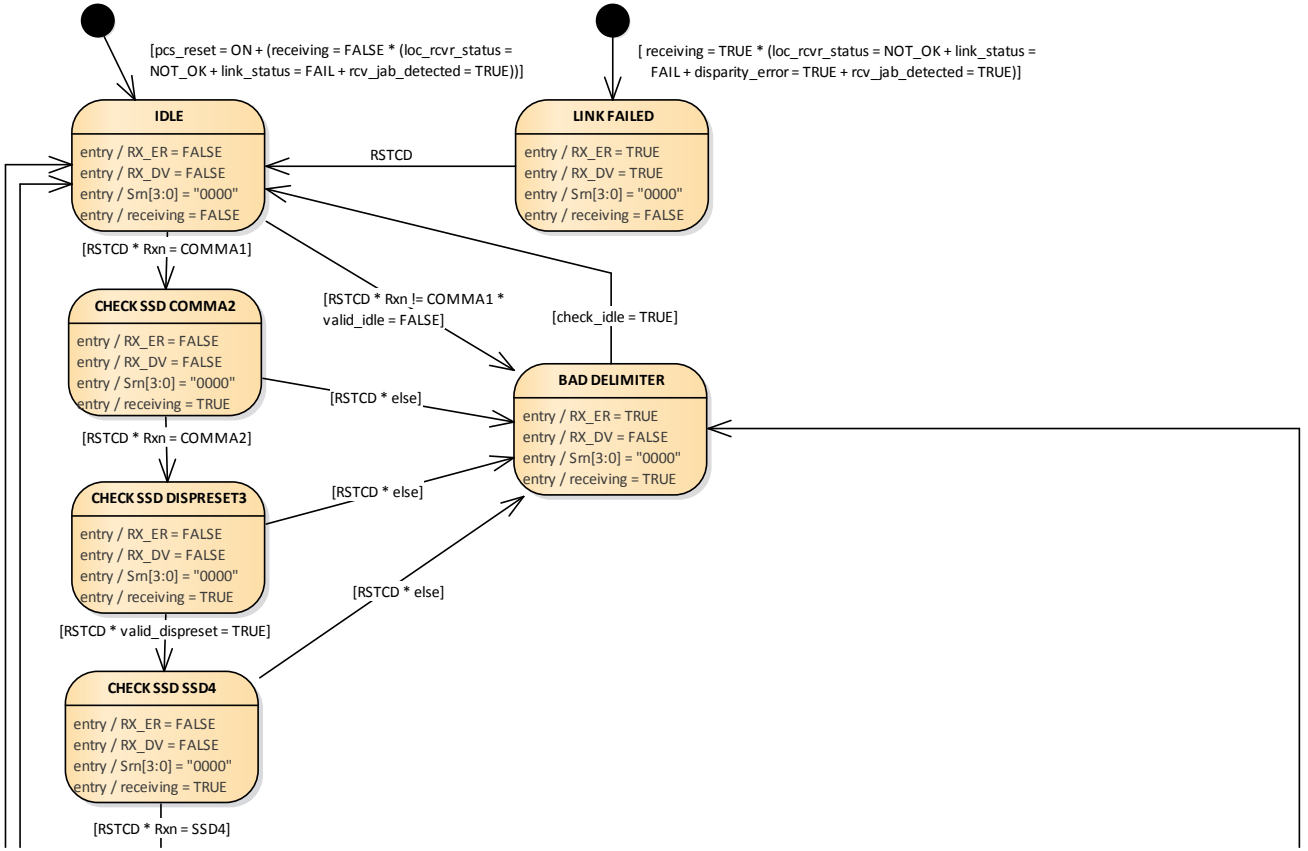
- The following block diagram shows an overview of the symbol decoding at the receiver side, which is done within the PCS:



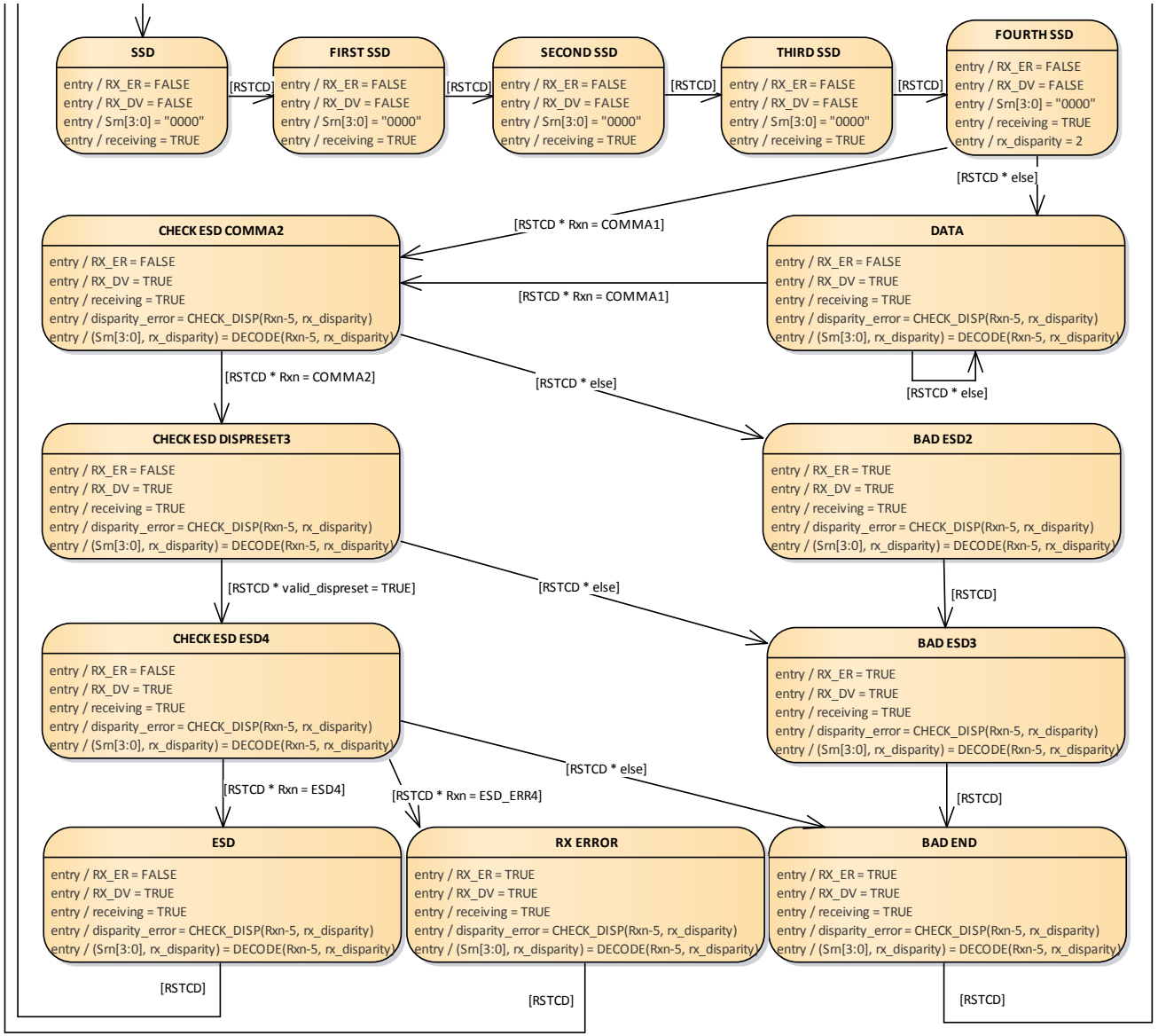
- The receive state machine on the next slide is shown in the center of the block diagram and used to control the different function blocks, depending on the control signals coming from the PMA layer or being created within the PCS layer.
- Also the RX_DV and RX_ER signals of the MII interface are created by the receive state machine.

Receive State Machine

- The receive state machine has two main states „IDLE“ and „DATA“.
- RSTCD is an abbreviation for „Receive Symbol Triplet Conversion Done“ and is synchronized with the PCS receive clock.
- When being in „IDLE“ state, as soon as the first COMMA1 symbol (0, 0, 0) is received, the „IDLE“ state is left and the decoding of the comma sequence (SSD) takes place.
- If the comma sequence can be decoded correctly, after filling the receive FIFO buffer the state machine goes into „DATA state.



Receive State Machine



Receive State Machine

- If there is a decoding error, the state machine goes into the „BAD DELIMITER“ state and waits until function „check_idle“ returns TRUE (e.g. after continuously receiving 8 IDLE triple ternary symbols).
- When being in „DATA“ state and receiving the first COMMA1 (0, 0, 0) symbol, the end of stream delimiter is being decoded, while the receive data are delivered from the receive FIFO buffer (this is the reason for the complexity of the state machine).
- If the ESD or ESD_ERR delimiters can be decoded correctly, the state machine goes back into „IDLE“ state, otherwise the state machine goes into state „BAD DELIMITER“.
- If an ESD_ERR or a wrong ESD sequence is being received the RX_ER output of the MII is being set.

- The DECODE function within the receive state machine decodes the received triple ternary symbols

$$Sr_n[3:0] = inverse\_table_{4B3T}(Rx_n)$$

- The second output value of this function is an updated rx_disparity value.
- This value is calculated in the following way:

$$rx\_disparity = rx\_disparity + disparity\ of\ currently\ received\ Rx_n$$

- The CHECK_DISP function checks, if the currently received triple ternary symbol is allowed for the current rx_disparity.

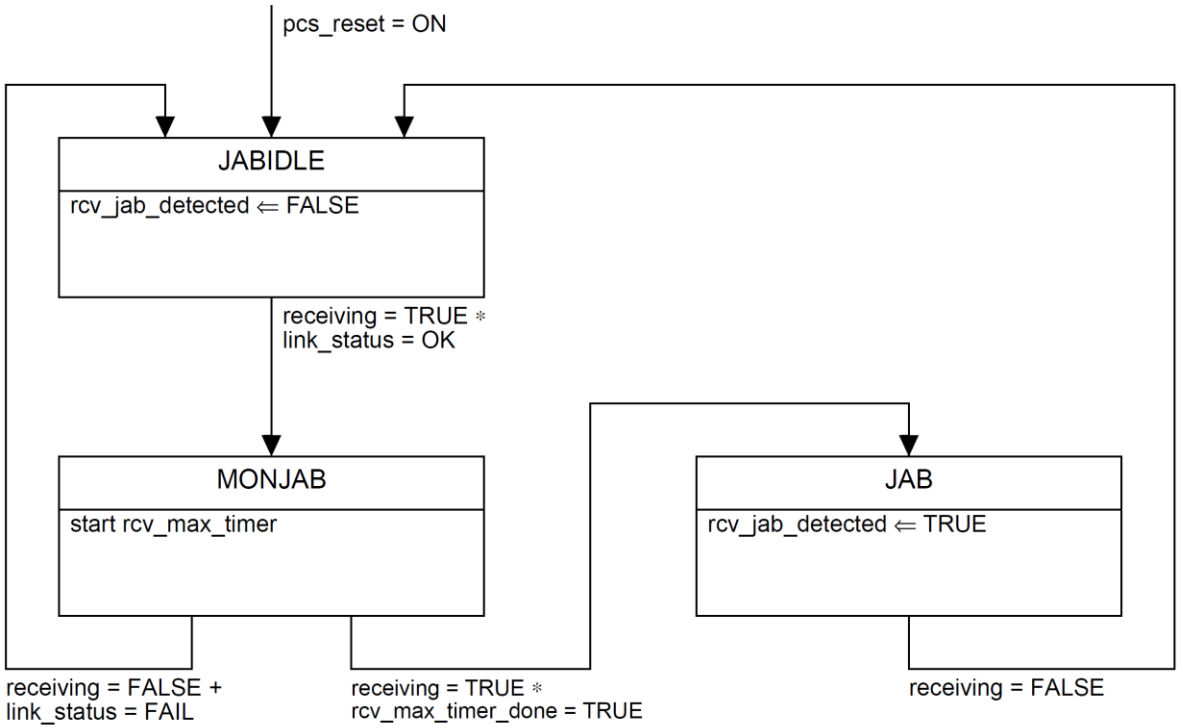
$$disparity\_error = (Rx_n\ !=\ table_{4B3T}(inverse\_table_{4B3T}(Rx_n), rx\_disparity)$$

- The valid_dispreset function returns TRUE, if one of the 4 allowed DISPRESET3 values is being received.

- The check_idle function returns TRUE, if e.g. 8 subsequent IDLE triple ternary symbols have been received.

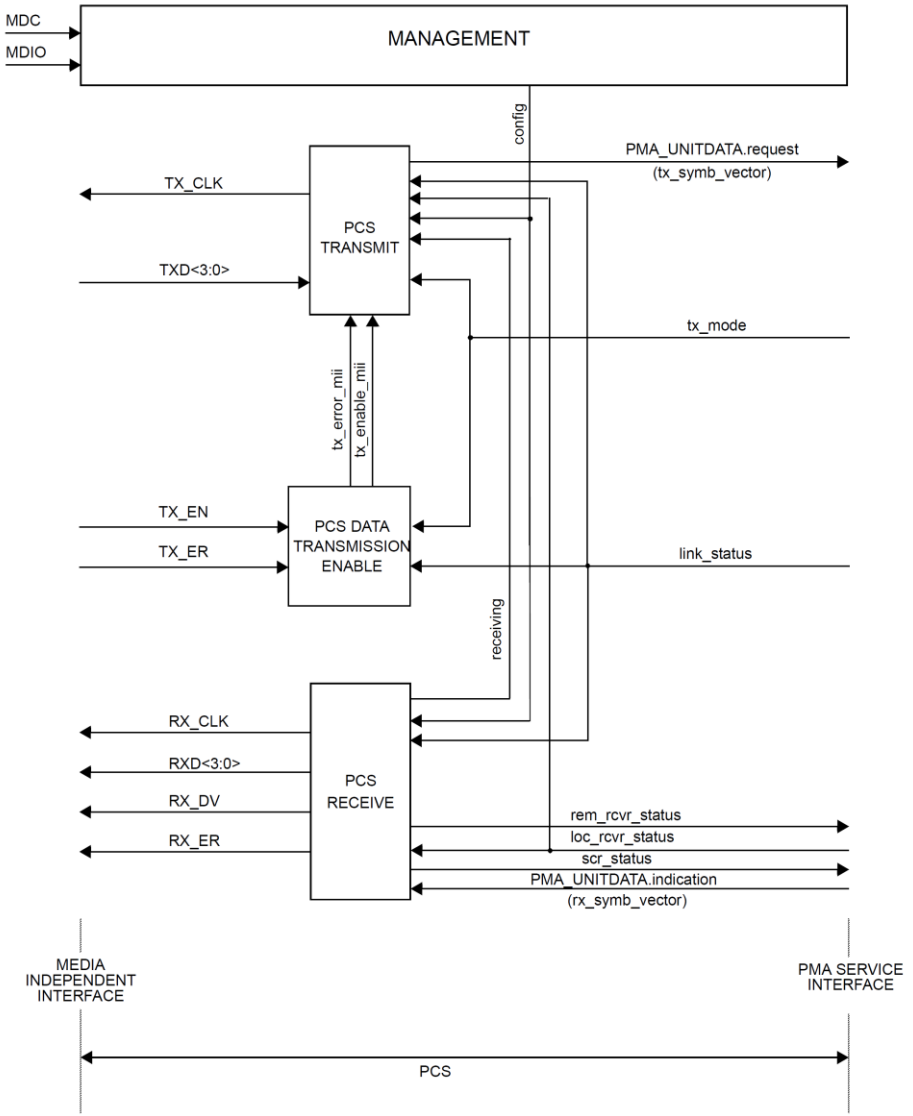
Jabber State Machine

- Additionally to the receive state machine a state machine for jabber detection needs to be implemented.
- In case an ESD (or ESD with error) is not being detected e.g. due to a receive bit error, this would cause the receive state machine to lock up in the DATA state.
- In this case (as receiving continuously stays „TRUE“), after expiration of the “rcv_max_timer”, the signal “rcv_jab_detected” is created, thus resetting the receive state machine.



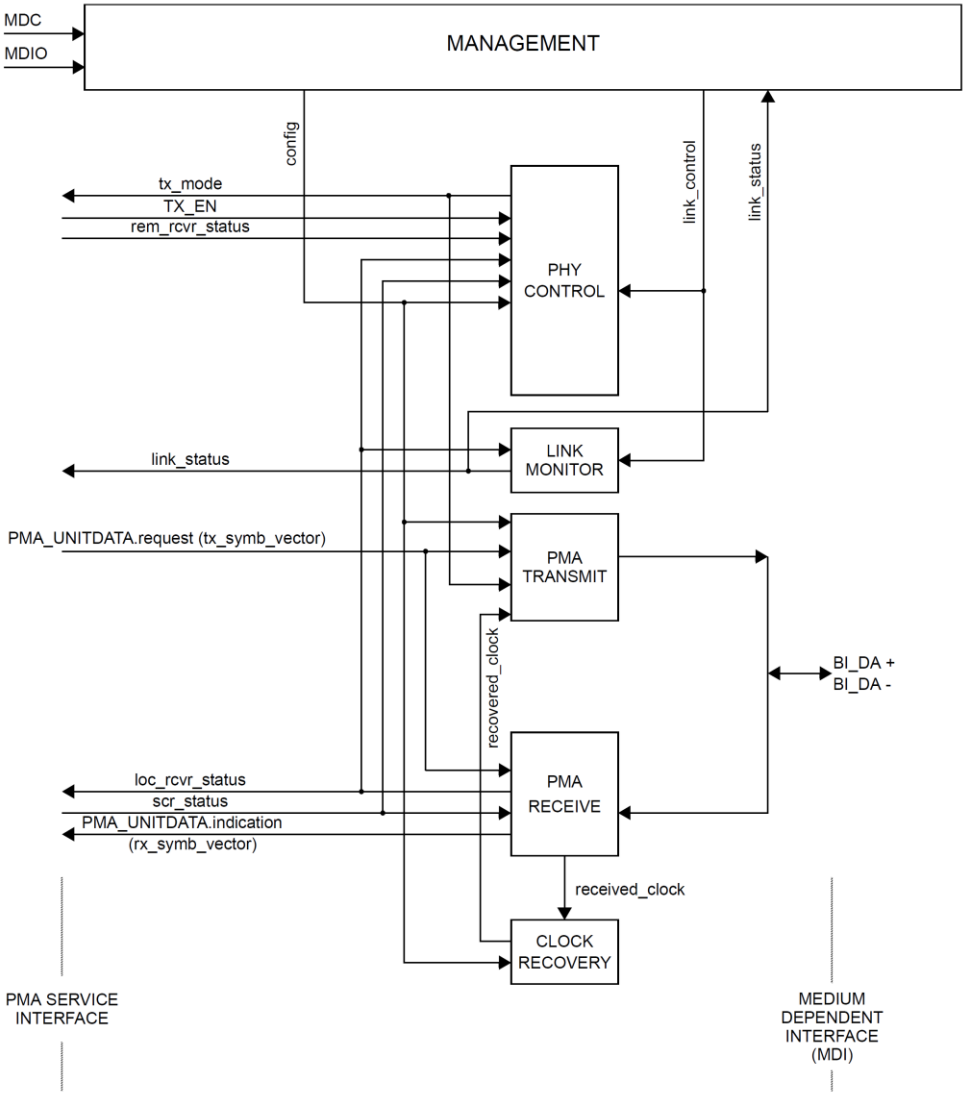
MII Interface

- As for other 10/100 Mbit/s Ethernet PHYs, a standard MII interface (Clause 22) is used.
- The four bit MII interface fits well to the 4B3T coding, as each nibble coming from the MII can be easily converted to the triple ternary symbol on the line.
- The MII interface allows an easy connection to existing microcontrollers.
- The specification of the MII interface does not preclude the implementation of other interfaces by a PHY manufacturer, but this is outside the scope of this document.



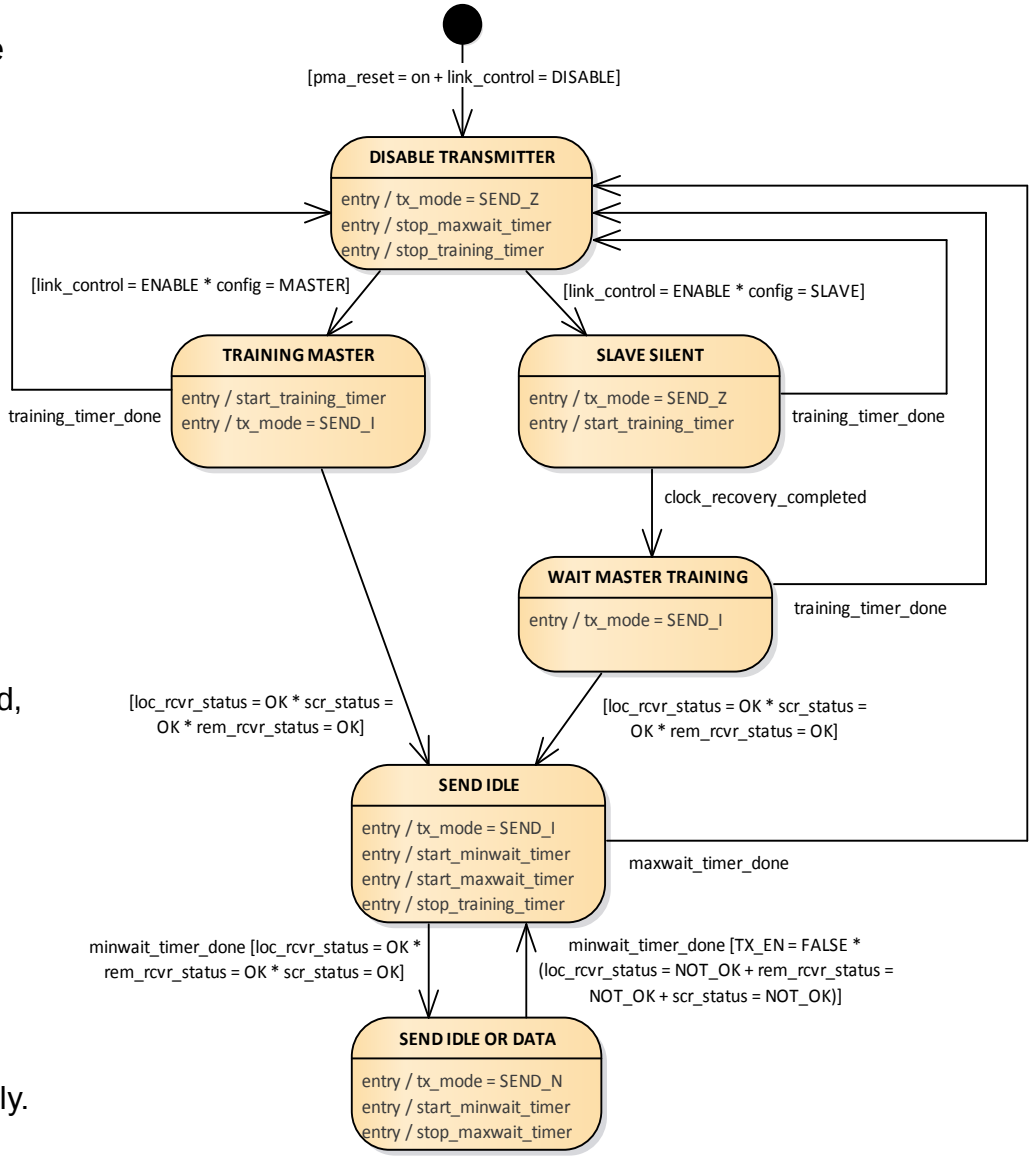
PMA Overview

- The PMA has two state machines, PHY Control state machine and Link Monitor state machine.
- PMA Transmit is used for sending the ternary symbols (tx_symb_vector) to the differential wire pair.
- PMA Receive is used for receiving the data from the differential wire pair and providing the ternary symbols to the PCS.
- A clock recovery block is necessary on the slave PHY side to synchronize the local oscillator to the oscillator of the master PHY.



PHY Control State Machine

- The PHY Control state machine is used to bring the PHY into the 10BASE-T1L mode of operation so that frames can be exchanged with the link partner.
- It determines whether the PHY operates in the normal mode, enabling data transmission over the link segment, or whether the PHY sends idle data, or disables the transmitter.
- There are two startup sequences, depending on which training time is needed during the startup.
- If there is no predetermined configuration available, the maximum time, until link_status = OK is reached, shall be less than 3000 ms (suggested value).
- If there is a predetermined configuration available (a set of valid filter coefficient is available), the maximum time from power_on = FALSE to link_status = OK shall be less than 100 ms.
- The timeout for the „training_timer“ is set accordingly.



PHY Control State Machine

- The slave PHY stays in SLAVE SILENT state until the recovered clock is stable, so that the slave PHY can enable its transmitter to send idle data. Depending on the PHY implementation this can help to speed up the training process, as the slave PHY can already start the transmission before the equalizer training is finished. Alternatively the slave PHY can wait until loc_rcvr_status = OK, but this will add some extra time needed for the filter training within the slave PHY

- After a disturbance on the link segment, e.g. when the current consumption on a powered link segment is quickly changed, the PHYs may not immediately drop the link, but need to try to recover the link for some time, before doing a complete restart. Therefore the “maxwait_timer” allows the PHYs to stay for some time in the SEND IDLE state before going to the DISABLE TRANSMITTER state.

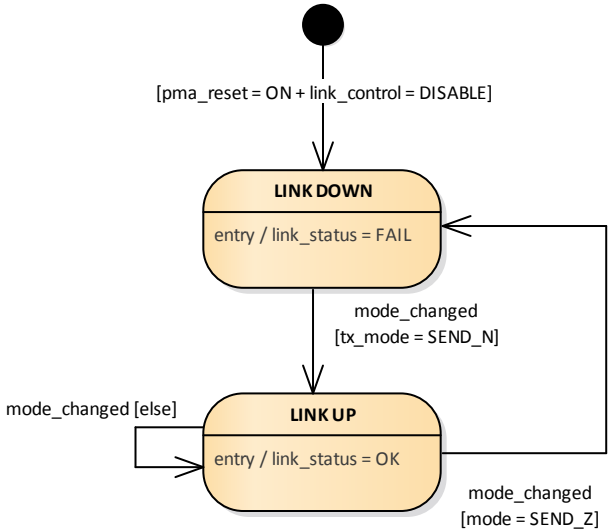
- See also presentation „http://www.ieee802.org/3/10SPE/public/adhoc/Graber_10SPE_06a_1016.pdf“ for some background information, why this behavior is needed.

- A time of 200 ms for the „maxwait_timer“ is suggested, to have some margin and compared to the training time of up to 3000 ms, this time is still quite short.

- Fast startup is expected to be limited to a very short link segment length only, where a predetermined configuration is available and where the change in ambient conditions has no significant influence on the available predetermined configuration or filter coefficient set.

Link Monitor State Machine

- The Link Monitor state machine generates the link status of the PHY.
- As soon as the PHY Control state machine sets the transmit mode to SEND_N (normal data transmission, both PHYs are synchronized and trained), the Link Monitor state machine sets the link status to ok.
- For the link status to stay ok it is accepted, that the PHYs temporarily go into SEND_I (sending idle packets for synchronization or training purposes).
- Only if the PHYs go into SEND_Z mode (disabling of the transmitter), after the „maxwait_timer“ has expired, the link status is set to fail and a complete retraining of the PHYs is started.



Transmitter Test Modes

- The following two transmitter test modes (already adopted in New Orleans) are used for verifying the electrical specifications of the transmitter:

  a) Test mode 1 – Transmitter output voltage, timing jitter, rise and fall times test mode
  b) Test mode 2 – Transmitter output droop test mode

- When test mode 1 is enabled, the PHY repeatedly transmits the data symbol sequence (+1, -1).
- When test mode 2 is enabled, the PHY repeatedly transmits ten “+1” symbols followed by ten “-1” symbols.

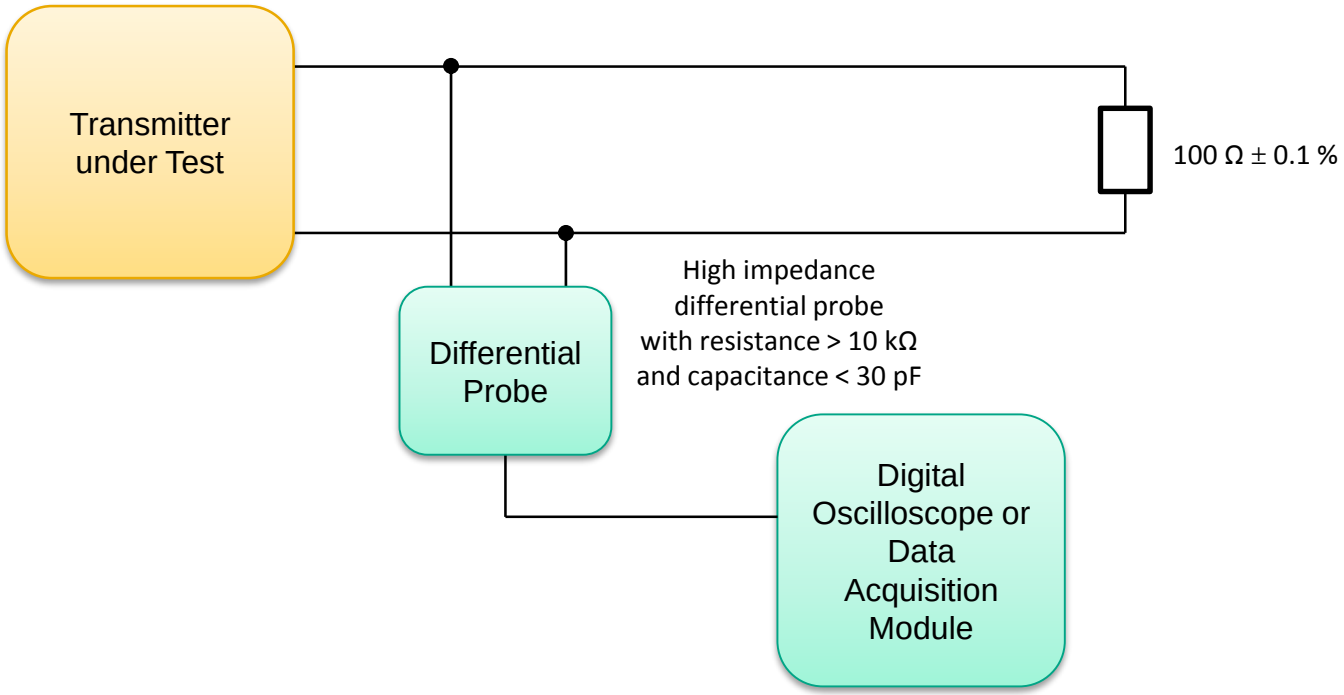
- The test modes can be enabled by setting the appropriate bits in the 10BASE-T1L PMA/PMD Test Control Register of the PHY Management register set.

- If MDIO is not implemented a similar functionality shall be provided by another interface.

- These test modes shall only change the data symbols provided to the transmitter circuitry and shall not alter the electrical and jitter characteristics of the transmitter and receiver from those of normal (non-test mode) operation.

Transmitter Test Fixture

- The following test fixture is used for performing the transmitter tests (already adopted in New Orleans):



Transmitter Electrical Specifications

- Transmitter output voltage (tested using test mode 1)
  - PAM-3 signaling (already adopted in New Orleans).
  - 2.4 V_{pp} (-1.2 V, 0 V, +1.2 V) ± 5 % using normal driving level (already adopted in New Orleans).
  - 1.0 V_{pp} (-0.5 V, 0 V, +0.5 V) ± 5 % using reduced driving level (already adopted in New Orleans).
  - Auto-negotiation of driving level.
  - Fixed selection of driving level possible over MDIO, similar functionality interface or optional bootstrap option.
  - Using smaller driving levels reduces the power consumption of the PHY and allows the use of the PHY in intrinsically safe link segments, but also reduces the maximum possible link segment length. A transmitter output amplitude of 1 V_{pp} instead of 2.4 V_{pp} reduces the transmit power of the PHY by approx. 8 dB, thus allowing only a shorter link segment length assuming the same noise tolerance.

- Transmitter output droop (tested using test mode 2)
  - Suggested is a maximum allowed droop of 20 % (taking the inner 9 bit times of the 10 bit times pulse length of the waveform specified in test mode 2).
  - As the maximum pulse length in a 4B3T code is 5 bit times, alternatively a shorter pulse length could be defined (adopting the droop limit).
  - Using a pulse length of 10 bit times likely reduces the measurement tolerances compared to a shorter pulse length.

- Transmitter timing jitter (tested using test mode 1)
  - Currently a maximum symbol-to-symbol jitter of ± 10 ns has been adopted.
  - Further investigation is necessary, if this value needs to be reduced to a smaller limit.

Transmitter Electrical Specifications

- Transmitter rise and fall times (tested using test mode 1)
  - Suggested is a rise time of the signal of 53.333 ns ± 10 % for a -1 to +1 transition.
  - Suggested is a fall time of the signal of 53.333 ns ± 10 % for a +1 to -1 transition.
  - As the rise and fall times are specified between 10 % and 90 % of the signal amplitude, assuming a linear transition between 0 % and 100 %, this equals to half of a symbol time (suggested value).
  - Below 10 % and above 90 % of the signal amplitude a lower slew rate is acceptable, to consider the lower speed of the output driver when operating near to its supply rails.

- Transmit clock frequency
  - Transmit symbol clock is 7.5 MHz (already adopted in New Orleans).
  - Transmit clock tolerance is ± 50 ppm (already adopted in New Orleans).

- Transmitter peak differential output
  - Specifies the maximum peak differential output voltage of the transmitter including the droop.
  - Currently suggested is a value of 2.76 V_{pp} for the normal driving level and 1.15 V_{pp} for the reduced driving level.
  - The limits are specified based on the suggested limit curves for the droop test.
  - As for a 4B3T code the maximum pulse length is 5 bit times, we could think about reducing these limits.

Receiver Electrical Specifications

- Bit Error Rate
  - The specified bit error rate (BER) for a long reach link segment is 10^{-9} as defined within the objectives.
- Receiver frequency tolerance
  - Receive symbol clock is 7.5 MHz (already adopted in New Orleans).
  - Receive clock frequency tolerance is ± 50 ppm (already adopted in New Orleans).
- Alien crosstalk noise rejection
  - Currently 2 tests are suggested.
  - The first test is using an AWGN disturber.
  - The second test is using an additional PHY configured in master mode in conjunction with a resistive coupling network.
  - Details can be seen in presentation „http://www.ieee802.org/3/cg/public/Sept2017/Graber_3cg_14_0917.pdf“.
  - It needs to be discussed, if these tests make sense, or if there are better tests for the receiver.

Delay Constraints

- Especially when thinking about having a lot of point-to-point link segments connected together within a redundant ring structure, maximum delay times for the transmit and receive path within the PHY chip need to be defined.

- Additional delay constraints will likely apply when thinking about TSN networks.

- Currently suggested is a delay time of maximum 32 bit times (3.2 µs) for the transmit path and 64 bit times (6.4 µs) for the receive path.

Management Interface

- It is suggested to use an optional MDIO management interface according to Clause 45.
- If no MDIO management interface is available, a similar interface needs to be implemented for PHY configuration purposes.

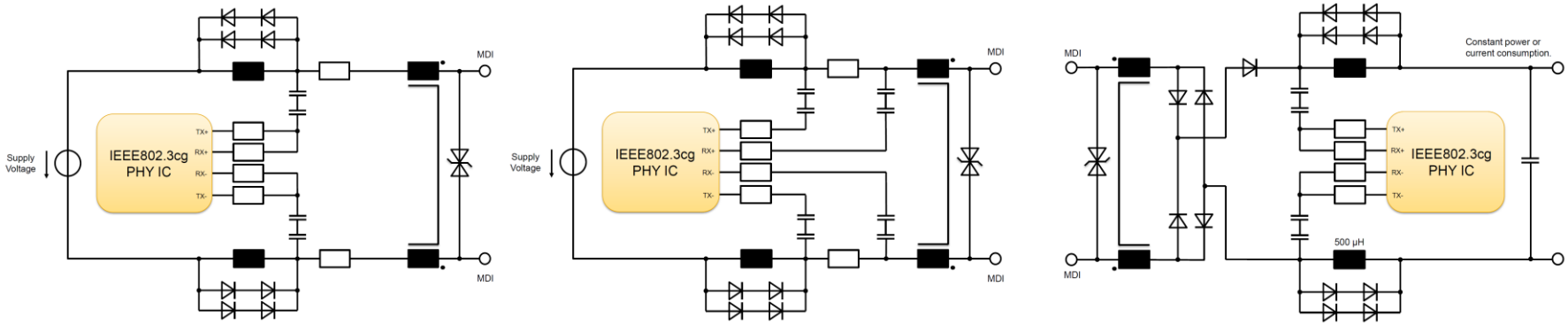
- The management interface currently specifies the following functionality:
  - PCS Reset
  - Optional PCS Loopback
  - PMA Reset
  - Optional PMA Loopback
  - Link Status
  - Master/Slave Mode
  - Transmit Driving Levels
  - Auto-Negotiation Settings
  - Test Mode Settings

- For relevant functions also a bootstrap option/pin configuration of the PHY IC in addition to the MDIO makes sense.

Informative Annex for Intrinsic Safety

- Intrinsically safe systems can be based on a 10BASE-T1L PHY, if the constraints described within the informative annex are taken into account.
- When using a PHY IC within a PHY intended for the use in intrinsically safe applications, external termination resistors, outside of the PHY IC are recommended to be able to adopt the termination resistors in accordance with the used intrinsically safe concept and to limit the energy to/from the intrinsically safe link segment in case of a failure.
- This would mean, that for such applications, separate pins for the differential transmit outputs and the differential receive inputs are recommended within a PHY IC.
- The differential receive inputs should be high impedance, to allow external series resistors, to limit the energy being provided from/to the receive inputs to/from the intrinsically safe link segment in case of an internal failure of the PHY IC.
- Further information about how to use a 10BASE-T1L PHY within intrinsically safe applications can be found in IEC 60079-0, IEC 60079-11 and other relevant national and international standards.

- The following schematics show in principle two implementations of a PSE circuit (left and middle figure) and one version of a PD circuit (right figure):



Thank You