Mapping Point To Point Protocol Over The Entire SONET/SDH SPE Or Over Sub Rate Tributary SPE's.

Preliminary Issue 1: June, 1996

TABLE OF CONTENTS

OVERVIEW

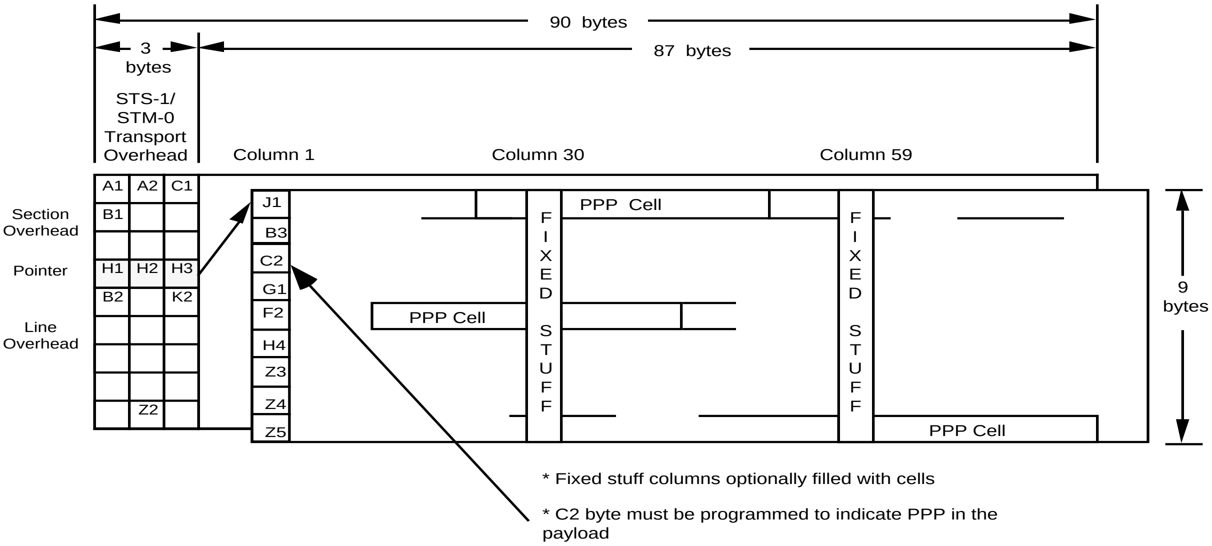
Introduction

This application note addresses the need to transfer a point to point protocol (such as the Internet Protocol, IP, or any other packet based PPP) over a SONET/SDH payload envelope or the lower rate tributary structures mapped inside the STS-12/STM-4, STS-3/STM-1(AU3)or STS-1/STM-0 frame structures. This functionality is easily implemented, as described in this document, using the PMC-Sierra Stel/ar chipset plus some external glue logic. The Stel/ar chipset consists of the PM5312 (STTX), PM5343 (STXC), PM5344 (SPTX) and the PM5362 (TUPP-PLUS). The external glue logic therefore interfaces between the SPTX and the PPP Assembler/Disassembler. For the case of a tributary (VT or TU) based solution, the TUPP-PLUS is utilized and very similar external glue logic would be interfaced between the TUPP-PLUS and the PPP cell Assembler/Disassembler. Both of these schemes are fully described in this document.

Transportation of PPP over the higher rate concatenated SONET/SDH payloads can be accomplished using the S/UNI-PLUS or the S/UNI-622. This is described in a separate application note described in PMC-Sierra document number PMC-960724.

Frame Structures Over Which PPP is Deployed Using The Stel/ar Chipset

Figure 1 shows the STS-1/STM-0 mapping. Fixed stuff bytes exist at columns 30 and 59 and cannot be used for payload. The remaining payload can be used entirely for the PPP data.



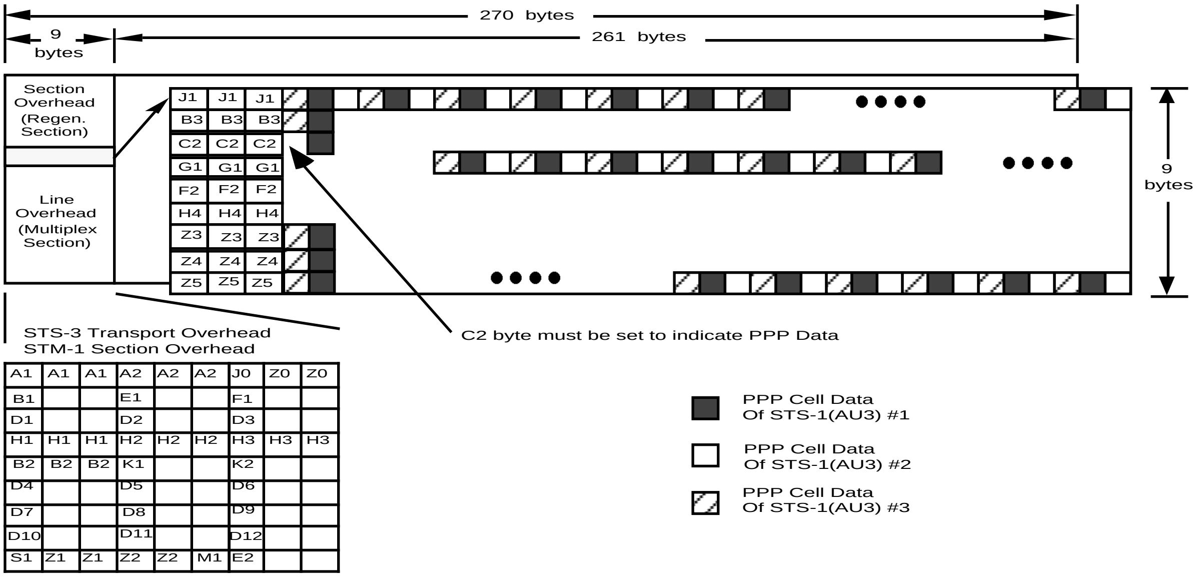
* Fixed stuff columns optionally filled with cells

* C2 byte must be programmed to indicate PPP in the payload

Figure 1: A SONET/SDH STS-1/STM-0 Frame Showing Transport Overhead

Alternatively, the SPE can be configured to carry SONET VT's or SDH TU's. In this case the mapping of the PPP data must be done at the VT/TU level using the TUPP-PLUS.

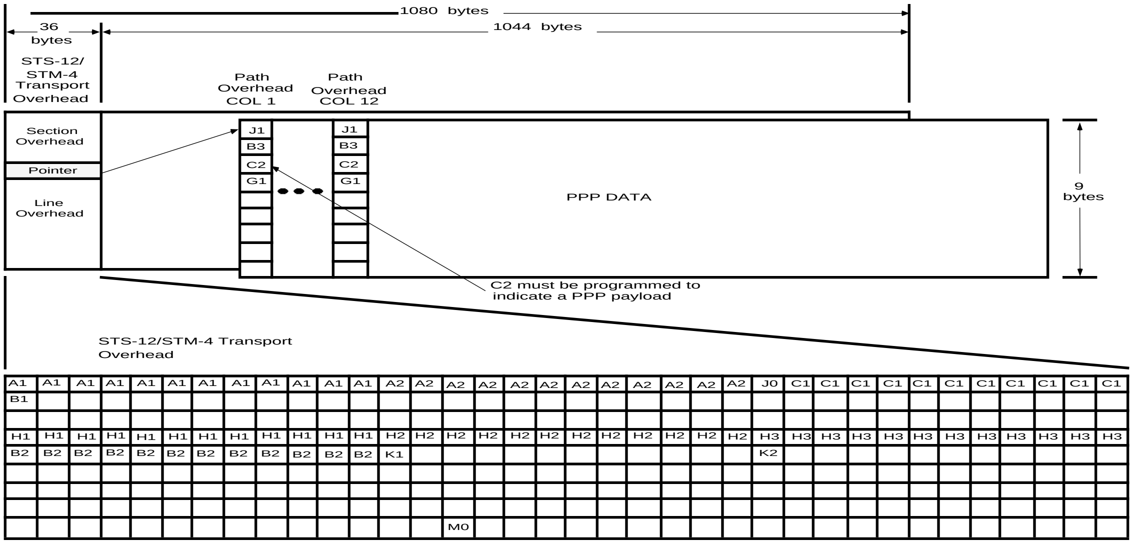
Figure 2 shows the STS-3 (STM-1) mapping. Stuff columns within each of the three STS-1's within the STS-3 signal are not shown but exist at the same locations as shown in figure 1. These stuff columns cannot be used for PPP payload. The remaining payload can be used entirely for the PPP data. Again, the SPE can be configured to carry SONET VT's or SDH TU's. In this case the mapping of the PPP data must be done at the VT/TU level using the TUPP-PLUS.


270 bytes
9 bytes
261 bytes
Section Overhead (Regen. Section)
Line Overhead (Multiplex Section)
J1 J1 J1
B3 B3 B3
C2 C2 C2
G1 G1 G1
F2 F2 F2
H4 H4 H4
Z3 Z3 Z3
Z4 Z4 Z4
Z5 Z5 Z5
9 bytes
STS-3 Transport Overhead
STM-1 Section Overhead
C2 byte must be set to indicate PPP Data
■ PPP Cell Data Of STS-1(AU3) #1
□ PPP Cell Data Of STS-1(AU3) #2
▨ PPP Cell Data Of STS-1(AU3) #3

A1	A1	A1	A2	A2	A2	J0	Z0	Z0
B1			E1			F1		
D1			D2			D3		
H1	H1	H1	H2	H2	H2	H3	H3	H3
B2	B2	B2	K1			K2		
D4			D5			D6		
D7			D8			D9		
D10			D11			D12		
S1	Z1	Z1	Z2	Z2	M1	E2		

Figure 2: A SONET/SDH STS-3/STM-1(AU3) Frame Showing Transport Overhead

Figure 3 shows the STS-12 (STM-4) mapping. Again, the stuff columns within each of the twelve STS-1's within the STS-12 signal are not shown but exist at the same locations as shown in figure 1. These stuff columns cannot be used for PPP payload. The remaining payload can be used entirely for the PPP data. Again, the SPE can be configured to carry SONET VT's or SDH TU's. In this case the mapping of the PPP data must be done at the VT/TU level using the TUPP-PLUS.



A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A1	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	A2	J0	C1	C1	C1	C1	C1	C1	C1	C1	C1	C1	C1
B1																																			
H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H1	H2	H2	H2	H2	H2	H2	H2	H2	H2	H2	H2	H2	H3	H3	H3	H3	H3	H3	H3	H3	H3	H3	H3	H3
B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	B2	K1												K2											
															M0																				

Figure 3: A SONET/SDH STS-12/STM-4 Frame Showing Transport Overhead

By decoding a signal that identifies the payload bytes it is possible to extract or map PPP data into the SONET/SDH SPE's of each of the frame structures discussed above. This application note describes the external logic required to identify the PPP payload bytes within a SONET/SDH SPE and SONET/SDH tributaries at the DROP and ADD interface.

SYSTEM ARCHITECTURE'S FOR TRANSPORTING PPP

Four options exist for transporting PPP data using the Stel/ar chipset. The first two architectures cater to the STS-3/STM-1(AU3)and STS-1/STM-0 rates, while the remaining two architectures cater to the higher STS-12/STM-1(AU3)rate. Higher rates than STS12/STM-4 can be accommodated using modified versions of these architectures, however, these are not described in this application note.

1: PPP Mapped Over an STS-3 or STS-1 SONET/SDH SPE.

The PM5344 (SPTX) contains a COMBUS type ADD and DROP interface, consisting of the DC1J1V1 and DPL control signals on the DROP bus and the AC1J1V1 and APL on the ADD bus. By using these signals to decode the SONET payload envelope bytes it is easy to extract/insert PPP data to/from the PPP Cell Assembler/Disassembler. Figure 4a and 4b below shows the system architecture required for implementing PPP over an STS-3/STM-1(AU3) or STS-1/STM-0 rate. The 155.52MHz/51.84MHz front end and SONET/SDH path processing can be implemented using the PMC-Sierra PM5343 (STXC), the PMC-Sierra PM5344 (SPTX) and some external clock and data recovery circuit. The DMA Controller passes the data bytes to/from memory to the HDLC Controller. The RFC-1619 HDLC Controller consists of an HDLC Packet Assembler function for the transmit direction and an HDLC Packet Disassembler function for the receive direction.

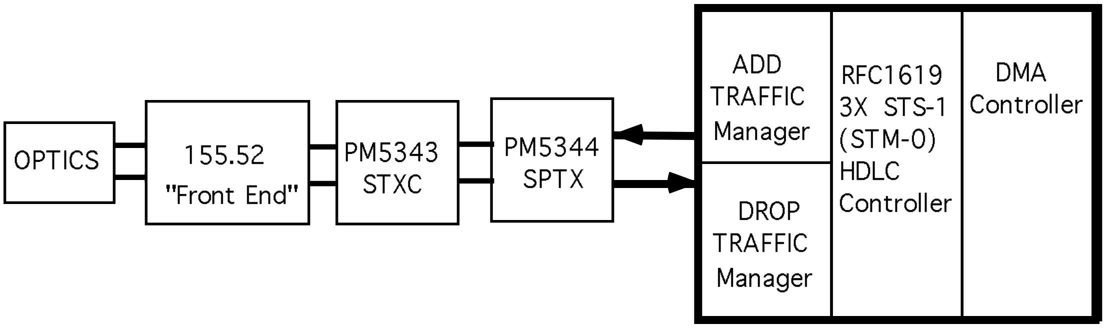


Figure 4a: PPP over an STS-3 /STM-1(3 x AU3) Link

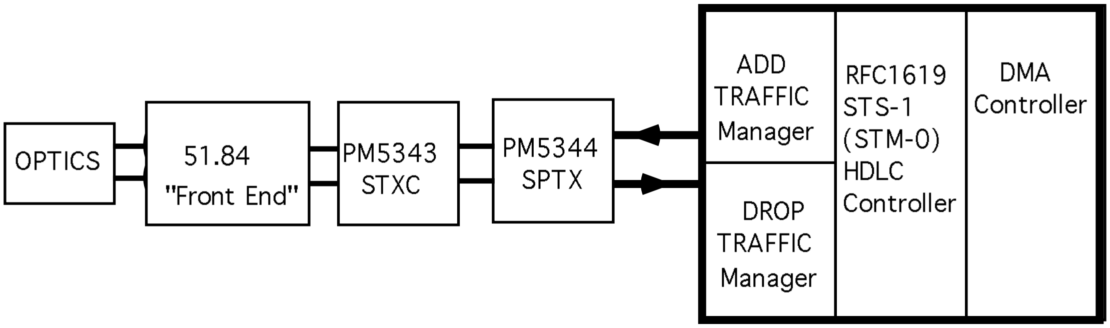


Figure 4b: PPP over an STS-1/STM-0(AU3) Link

In the transmit direction, the ADD Traffic Manager transfers the PPP data from the HDLC Packet Assembler to the SPTX. The PPP data bytes are transferred under the control of the decoded payload byte identification signals. Each payload PPP byte is identified by decoding the arbitrarily aligned AC1J1V1 and APL inputs to the SPTX.

In the receive direction, the DROP Traffic Manager transfers the PPP data from the SPTX to the HDLC Packet Disassembler. The PPP data bytes are transferred under the direction of the decoded SPE payload byte identification signals. Where each payload PPP byte is identified by decoding the DC1J1V1 and DPL outputs of the SPTX's Drop bus.

This application note will describe the required decoding logic in the DROP Traffic Manager and the ADD traffic Manager.

2: PPP Mapped Over Lower Level Tributaries Within an STS-3 or STS-1 frame

This option is very similar to the first option. The main difference is that the ADD Traffic Manager and DROP Traffic Manager map and extract PPP bytes at the VT/TU level rather than at the SONET/SDH frame SPE level. Figure 5a and 5b show the architecture required to implement an STS-3/STM-1(AU3) or an STS-1/STM-0 link.

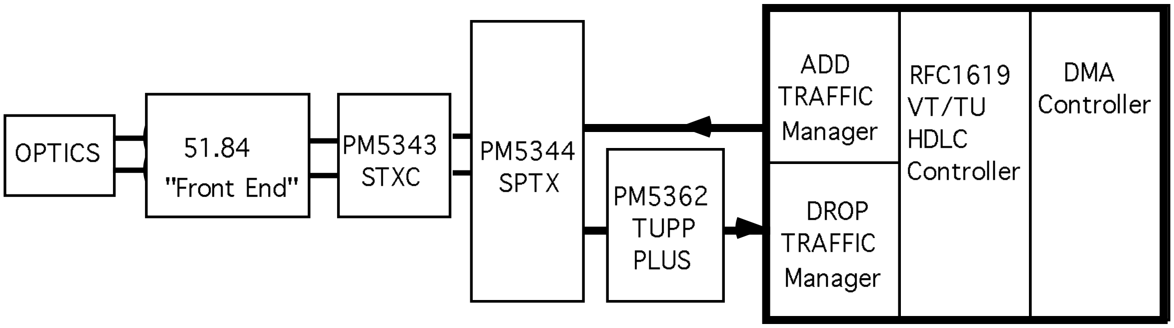


Figure 5a: Tributary Mapped PPP over an STS-1 /STM-0 (AU3)Link

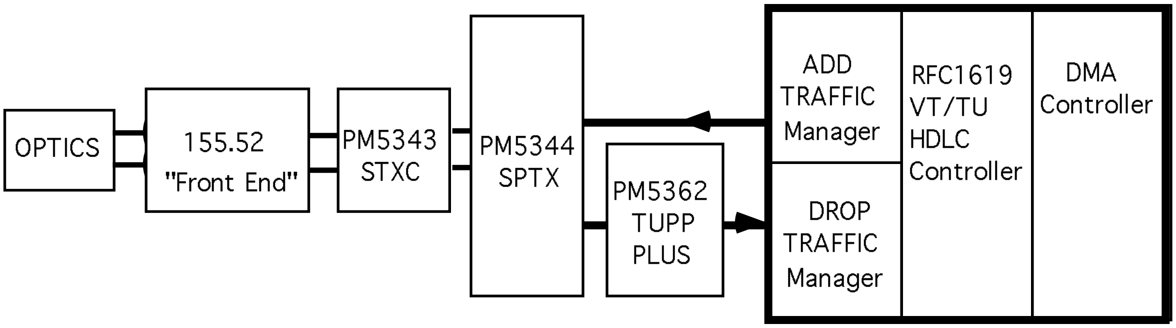


Figure 5b: Tributary Mapped PPP over an STS-3 /STM-1 (3 X AU3)Link

In the transmit direction, the ADD Traffic Manager transfers the PPP data from the HDLC Packet Assembler to the SPTX. The PPP data bytes are transferred under the control of the VT/TU level decoded payload byte identification signals. Each payload PPP byte is identified by decoding the arbitrarily aligned AC1J1V1 and APL inputs to the SPTX.

In the receive direction, the DROP Traffic Manager transfers the PPP data from the TUPP-PLUS to the HDLC Packet Disassembler. The PPP data bytes are transferred under the direction of the decoded VT/TU SPE payload byte identification signals. Where each payload PPP byte is identified by decoding the LC1J1V1 and LPL outputs of the TUPP-PLUS Drop bus.

This application note will describe the required tributary level decoding logic in the DROP Traffic Manager and the ADD traffic Manager.

3: PPP Over an STS-12(STM-4) SONET/SDH Frame.

Figure 6 below shows the system architecture for implementing PPP over an STS-12/STM-4 rate. The 622MHz front end and SONET path processing can be implemented using the PMC-Sierra PM5312 (STTX), the PMC-Sierra PM5344 (SPTX) and some external clock and data recovery circuit. The DMA Controller passes the bytes of data to/from memory to the HDLC Controller. The RFC-1619 HDLC Controller consists of an HDLC Packet Assembler function for the transmit direction and a HDLC Packet Disassembler function for the receive direction.

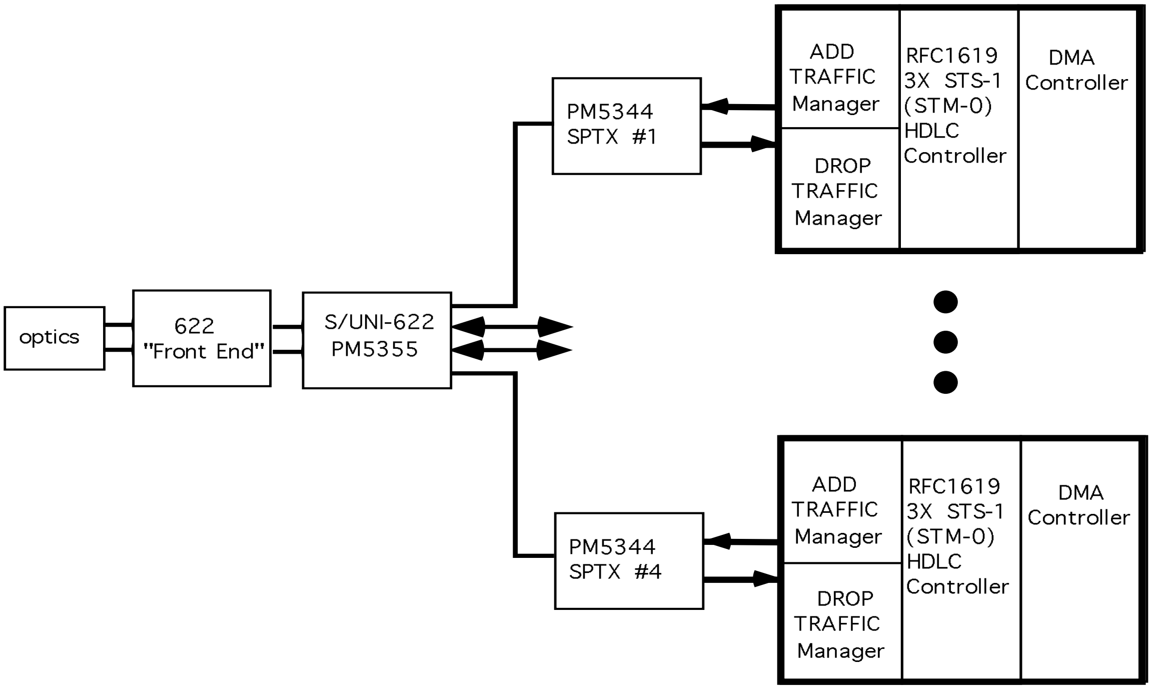


Figure 6: PPP over an STS-12/STM-4 Link

In the transmit direction, the ADD Traffic Manager transfers the PPP data from the HDLC Packet Assembler to the SPTX. The PPP data bytes are transferred under the control of the decoded payload byte identification signals. Each payload PPP byte is identified by decoding the arbitrarily aligned AC1J1V1 and APL inputs to the SPTX. The internal decoding logic of the ADD Traffic Manager blocks shown in figure 6 are exactly the same as the ADD Traffic Manager block shown in figure 4.

In the receive direction, the DROP Traffic Manager transfers the PPP data from the SPTX to the HDLC Packet Disassembler. The PPP data bytes are transferred under the direction of the decoded payload byte identification signals. Where each payload PPP byte is identified by decoding the generated DC1J1V1 and DPL outputs of the SPTX's Drop bus. The internal decoding logic of the DROP Traffic Manager blocks shown in figure 6 are exactly the same as the DROP Traffic Manager block shown in figure 4.

4: PPP Mapped Over Lower Level Tributaries Within an STS-12 frame

This option is very similar to the second option. The main difference being the 4 SPTX/TUPP-PLUS interfaces instead of just one.

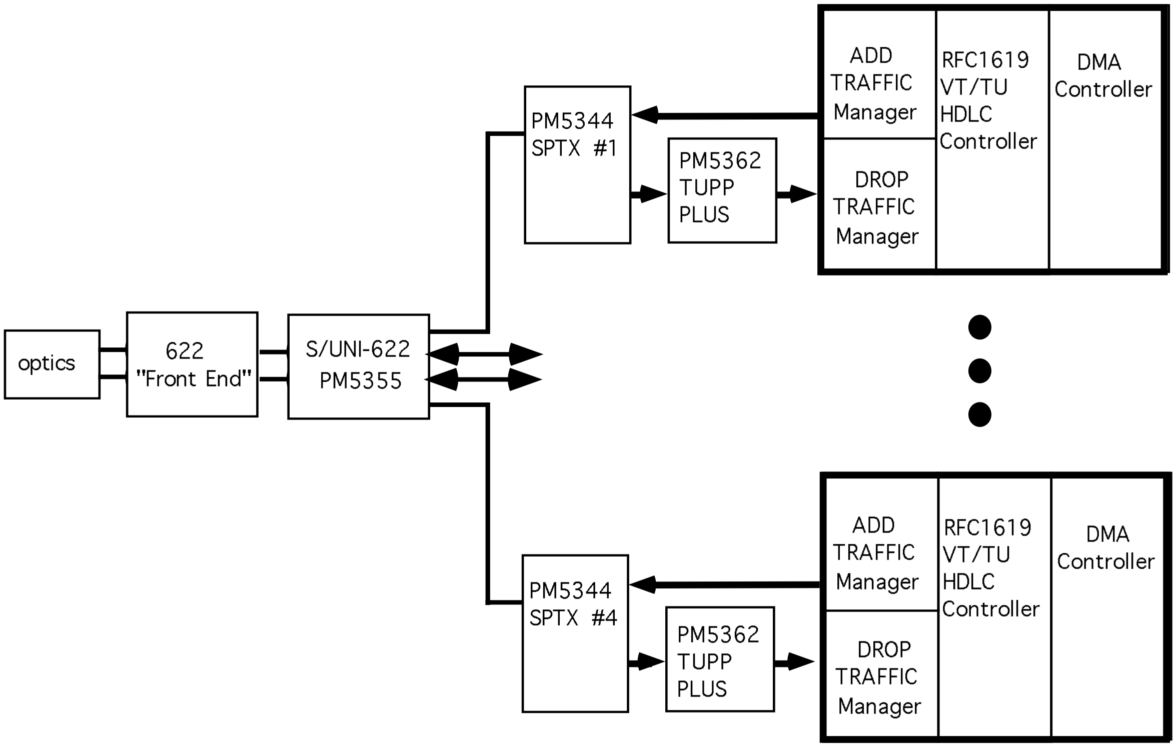


Figure 7a: Tributary Mapped PPP over an STS-12 Link

In the transmit direction, the ADD Traffic Manager maps the PPP data from the HDLC Packet Assembler to a tributary SPE at ADD bus of the SPTX. The PPP data bytes are transferred under the control of the VT/TU level decoded payload byte identification signals. Each payload PPP byte is identified by decoding the arbitrarily aligned AC1J1V1 and APL inputs to the SPTX.

In the receive direction, the DROP Traffic Manager transfers the PPP data from the TUPP-PLUS to the HDLC Packet Disassembler. The PPP data bytes are transferred under the direction of the decoded VT/TU SPE payload byte identification signals. Where each payload PPP byte is identified by decoding the LC1J1V1 and LPL outputs of the TUPP-PLUS Drop bus.

The tributary level decoding logic in the DROP Traffic Manager and the ADD traffic Manager are exactly the same as the logic required in option 2.

5: PPP Mapped Over an STS-48c/STM-16c frame

This option is unlike any of the previous options. The reason being that the HDLC controller must write the PPP bytes into the STS-3 rate streams according to the multiplexing stucture of an STS-48c. All the STS-3 rate streams at the ADD TRAFFIC Manager to I/F GLUE interface actually belong to a single path, and therefore one single SPE of size 48 X STS-1's. The HDLC controller must therefore write the PPP packets in an order such that the PPP packets appear contiguous in the STS-48c/STM-16c payload. This ordering must take into account the multiplexing behaviour of the STTX and the VSC8023 and VSC8034.

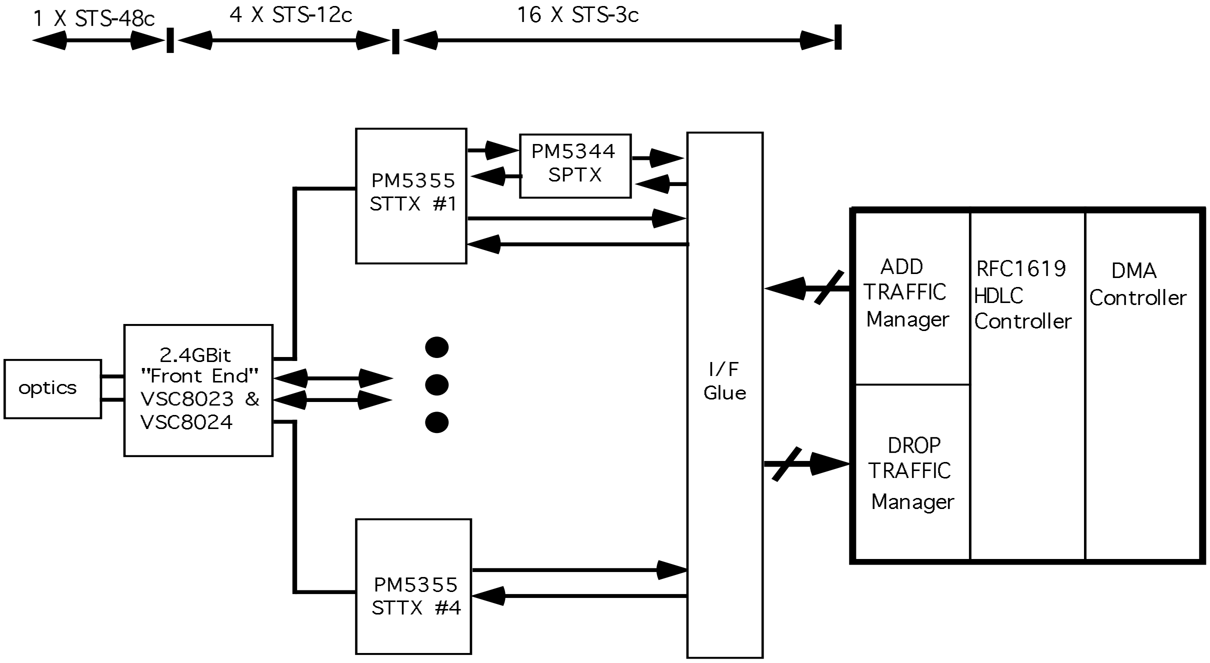


Figure 7b: PPP over an STS-48c Link

The interface between a single STTX and the I/F GLUE as well as the interface between the SPTX and the I/F GLUE is described in PMC-950730. That same logic can be extended to incorporate a further three STTX's as depicted in figure 7b and therefore this application note does not describe further details of this logic. Also, a stand alone application note for this option will be available at a future date.

IMPLEMENTATION DETAILS

In order to implement the four options described in the previous section, there are five interfaces that need to be defined in detail. These five interfaces include the interconnect between the PM5343 (STXC) and the PM5344 (SPTX), the interconnect between the PM5312 (STTX) and SPTX, the interface between the SPTX and the HDLC Controller, the interconnect between the SPTX and the PM5362 (TUPP-PLUS) and the interface between the TUPP-PLUS and the HDLC Controller.   Note, that the interfaces between the SPTX and the HDLC Controller and the TUPP-PLUS and the HDLC Controller will include the logic contained in the ADD Traffic Manager and DROP Traffic Manager Blocks.

STXC to SPTX Interface

This interface is very simple since these two devices are designed to snap together without any external glue logic. The basic interconnect is shown below in figure 8.

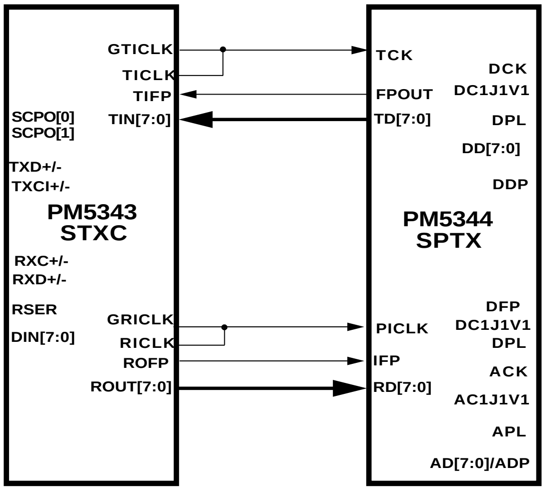


Figure 8: STXC To SPTX Interconnect

STTX to SPTX Interface

This interface requires 4 SPTX's and 1 STTX. It is important to synchronize all four SPTX's so that the four STS-3 rate streams can be multiplexed and demultiplexed by the STTX without glue logic. The interconnection between this interface is shown in detail in figure 9.

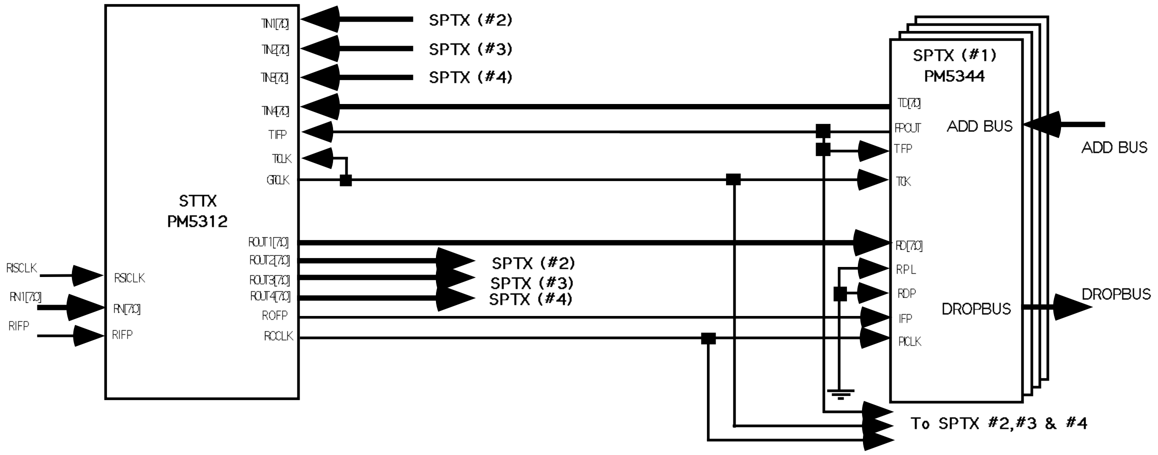


Figure 9: STTX To SPTX Interconnect

SPTX to TUPP-PLUS Interface

This interface is also very simple since these two devices are designed to snap together without any external glue logic. The interconnect is shown below in figure 10.

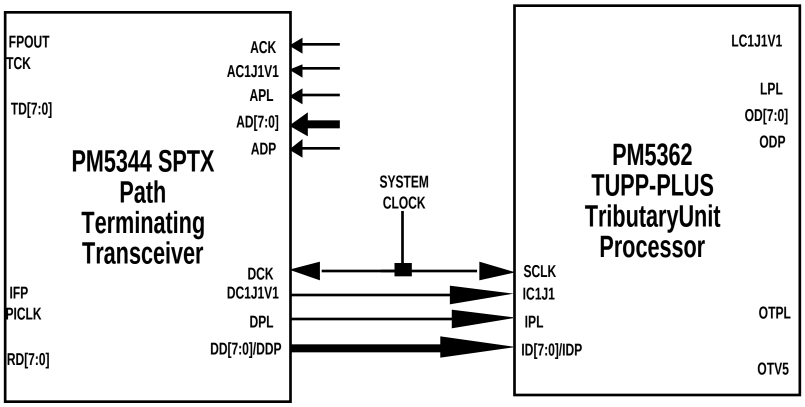


Figure 10: SPTX To TUPP-PLUS Interconnect

SPTX to HDLC Controller Interface

This logic interfaces to the ADD bus and the DROP bus. On the DROP bus, the outputs DC1J1V1 and DPL signals must be decoded to identify the SPE payload bytes from the path overhead and fixed stuff bytes. Note, the V1 part of DC1J1V1 must be turned off by programming the DISV1 bit in register 01H to logic 1. The ADD bus interface requires the AC1J1V1 and APL inputs. These inputs are generated by the ADD Traffic Manager logic as desired according to the arbitrarily aligned frame structure on the ADD bus. The mapping of PPP data bytes into the ADD bus SPE is implicit to the function that generates the AC1J1V1 and APL signals. Therefore no decoding is required when the HDLC Controller and the ADD bus mapper are one and the same. Decoding is required if the HDLC Controller is independent of the ADD bus mapper functionality. This document will describe the latter case.

DROP Traffic Manager (SONET/SDH Level)
Figure 11 shows the logic required to identify the PPP data bytes on the DROP bus interface. The SPE NUM DECODER modifies the DPL input to be valid during the appropriate STS-1 number to create the SPE#n outputs. The PPP BYTE DECODER further modifies the SPE #n outputs by masking out the fixed stuff bytes and the path overhead bytes (retimed to the falling edge) and gates this with the clock input to create the final output, PPP_SPE#n. PPP_SPE#n is therefore a gapped clock that clocks every time a valid PPP byte is received.

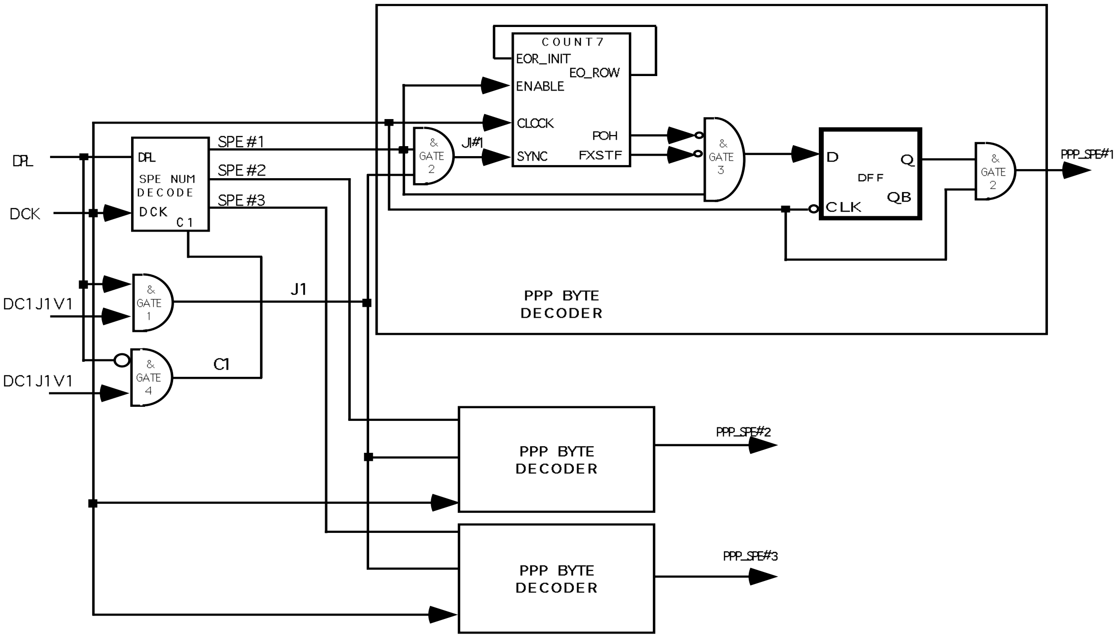


Figure 11: DROP Traffic Manager For The SPTX to HDLC Controller Interface

The DPL, C1 and DCK signals are used to decode SPE bytes for STS-1#1, STS-1#2 and STS-1#3. This is shown in figure 12 below and is discussed in the next section. The two input AND gate (&GATE1) decodes the J1 byte location on the DROP bus data output from the SPTX. The 2 input AND gate (&GATE2) decodes the J1 byte on the first STS-1 and synchronizes the 7 bit counter to a count of 0 on the next rising edge of clock. The two input AND gate (&GATE4) decodes the C1 byte location. The seven bit count (COUNT7) counts the SPE bytes in one row. Note, that because the counter is synchronized by the J1 byte indication and the DPL output is varied by ± 1 byte during a justified frame, the effect of pointer justifications on the decoding of "POH" and "FXSTF" will be automatically accounted for. The Counter has three outputs, EO_ROW, POH and FXSTF. EO_ROW is generated when the counter reaches a count of 86 (the 87th byte) and it reinitializes the counter to count from 0 on the next rising edge of clock. The FXSTF output indicates the fixed stuff byte location. These bytes are located at a count value of 29 and 59 (decimal). Similarly, the POH output is located at a count of 0. The three input AND gate (&GATE3) modifies the SPE#n input and gaps out the path overhead and fixed stuff columns. The final DFF retimes this to the falling edge and gates this with clock. The PPP_SPE#1 output is therefore a gapped clock of all bytes in which PPP data is valid. The rising edge of this clock should be used to sample a valid PPP byte.

For applications where only a 51.52MHz STS-1/STM-0 input is to be processed, the SPE NUM DECODE block, the &GATE4, the &GATE's inside PPP BYTE DECODER and the bottom two PPP BYTE DECODER blocks must be deleted. The ENABLE input of the counter will then connect directly to the DPL input and the SYNC input of the counter will connect to the output of &GATE1.

SPE NUM DECODER

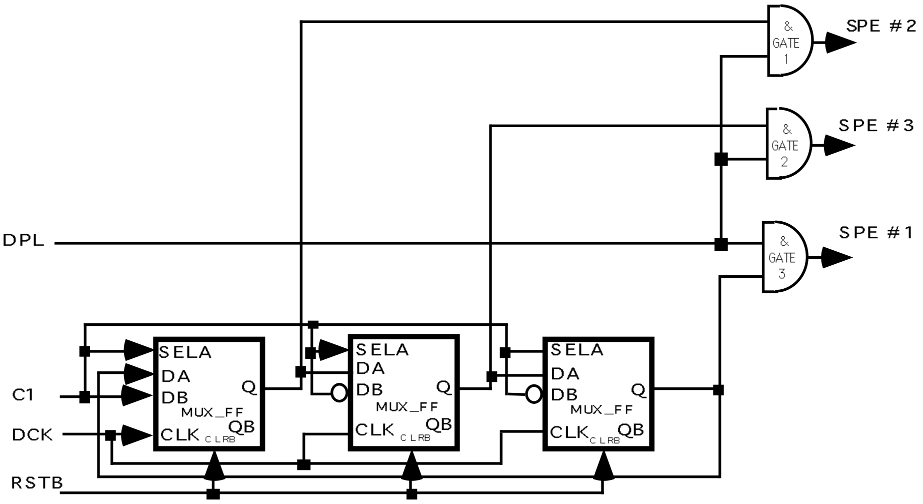


Figure 12: SPE Num Decoder

This block identifies the STS-1#1, STS-1#2 and STS-1#3 payload bytes. Normally, the three MUX-DFF's act as a 3 bit shift register, starting with a logic 1 clocked into the first MUX-DFF on the rising edge of clock. The circuit is initialized when C1 is logic 1. The SPE#n outputs are generated only when DPL is logic 1.

ADD Traffic Manager (SONET/SDH Level)

This block is in essence very similar to the functionality of the DROP Traffic Manager. However, for the case where the PPP HDLC controller is the generator of the AC1J1V1 and APL signals, it implicitly knows where the PPP data bytes reside. i.e. the functionality of the ADD Traffic Controller is swallowed up by the HDLC Controller. In such a case the implementation of the ADD Traffic Manager simply consists of two wires, one to signal AC1J1V1 and the other to signal APL from the HDLC Controller to the SPTX.

In the case the AC1J1V1 and APL signals are derived from, say, a delayed version of the DROP bus DC1J1V1 and DPL signals or from some other mapping master other than the HDLC Controller then the functionality of the ADD Traffic Controller is identical to the functionality shown for the DROP Traffic Controller. This is shown in figure 13 below.

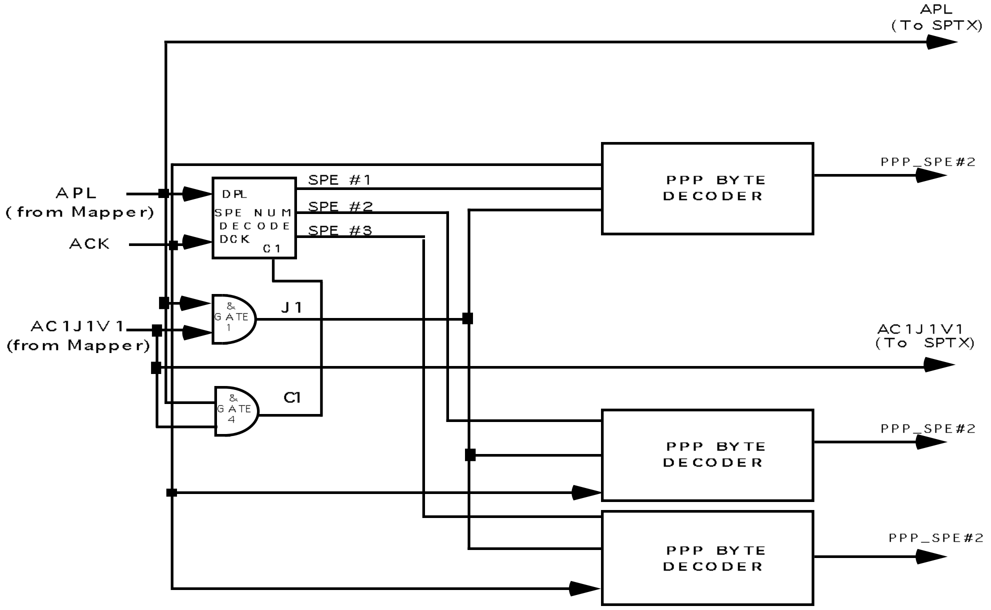


Figure 13: ADD Traffic Manager For The SPTX to HDLC Controller Interface

The SPE NUM DECODE block is the same as shown previously in figure 12 and the PPP BYTE DECODER block is the same as the one used in the DROP Traffic Manager.

For applications where only a 51.52MHz STS-1/STM-0 input is to be processed, the SPE NUM DECODE block, the &GATE4, the &GATE's inside PPP BYTE DECODER and the bottom two PPP BYTE DECODER blocks must be deleted. The ENABLE input of the counter will then connect directly to the APL input and the SYNC input of the counter will connect to the output of &GATE1.

TUPP-PLUS to HDLC Controller Interface

This interface consists of the ADD bus and the DROP bus. The TUPP-PLUS DROP bus, OTPL, OTV5, LC1J1V1 and LPL signals must be decoded to identify the tributary PPP payload bytes. Note, the V1 part of LC1J1V1 must be turned off by programming the DISV1 bit in register 01H to logic 1.

The ADD bus interface requires the AC1J1V1 and APL inputs. These inputs are generated by the ADD Traffic Manager logic as desired according to the arbitrarily aligned frame structure on the ADD bus. The mapping of PPP data bytes into the ADD bus tributaries is implicit to the function that generates the AC1J1V1 and APL signals. Therefore no decoding is required when the HDLC Controller and the ADD bus mapper are one and the same. Decoding is required if the HDLC Controller is independent of the ADD bus mapper functionality. This document will describe the latter case.

DROP Traffic Manager (Tributary Level)

This block is equivalent of the 'DROP Traffic Manager' used in the SPTX to HDLC Controller Interface. Valid tributary PPP bytes are indicated by the TPP_SPE#1, TPP_SPE#2 and TPP_SPE#3 outputs (generated by &GATE2, &GATE3 and &GATE4). The decoding of these signals is simple to implement using the LPL, OTPL and OTV5 outputs of the TUPP-PLUS, requiring only the SPE NUM DECODE block and &GATE1, &GATE2, &GATE3 and &GATE4. The rest of the circuitry (consisting of &GATE5 and the three TRIB ID DECODER blocks) is only required if a higher resolution of each tributary is needed by the HDLC Controller.

To identify each tributary, the gapped clock SPE1_TRIB_ID[n:0], SPE2_TRIB_ID[n:0] and SPE3_TRIB_ID[n:0] outputs are required. The SPE1_TRIB_ID[n:0] outputs identify the valid PPP bytes in each of the 'n' tributaries in the SPE of STS-1 #1 (or VC3 of an AU3 #1 in an SDH system). The SPE2_TRIB_ID[n:0] outputs identify the valid PPP bytes in each of the 'n' tributaries in the SPE of STS-1 #2 (or VC3 of an AU3 #2 in an SDH system). The SPE3_TRIB_ID[n:0] outputs identify the valid PPP bytes in each of the 'n' tributaries in the SPE of STS-1 #3 (or VC3 of an AU3 #3 in an SDH system).

TRIB ID DECODER

This block consists of a 7 bit counter (COUNT7) and a custom programmable block (CUSTOM TRIB ID). The COUNT7 block is enabled to count during all SPE data bytes, including path overhead bytes and any fixed stuff bytes using the output decoded by &GATE6. After counting for one complete row, the counter reinitializes itself (via the EO_ROW output generated at a count of 86 decimal) at the next rising edge to count again from zero. The counter is also initialized to a count of zero by the output of &GATE7 (indicating J1 byte) connected to the SYNC input of the counter. The CNT[7:0] output therefore indicates all SPE payload columns in the STS-1 payload envelope. *Note, that when the TUPP-PLUS is configured to operate in the locked mode of operation the three TRIB ID DECODER blocks can be collaped into a single counter since all J1's are adjacent to each other.*

The CUSTOM TRIB ID block is a general block that cannot be specifically defined in this document. The reason for this is that it can change from application to application, i.e. depending on whether the application uses VT1.5, VT2, VT3, VT6, TU11, TU12, TU2, or TU3 or some combination of these possibilities. The function of this block is simple and should be customized at the time of implementation. The functionality of this block is to use the CNT[6:0] count value and generate 'n' outputs (n depending upon the number of VT's/TU's being carried in the SONET:STS-1/SDH:AU3 SPE). Each SPE1_TRIB_ID[n] is a gapped clock active during the byte times relevant for the VT/TU required for identification. The GAPB input disables the clocking when the PPP bytes are not valid. For example, if tributary TU12 exists in columns 81, 144, 207 and 270, of an SDH/SONET frame then SPE1_TRIB_ID[3] will be active during CNT[6:0] count values 23,44, 65 and 86 and when GAPB is logic 1. Note, for reference, the J1 byte is located at a count of 0.

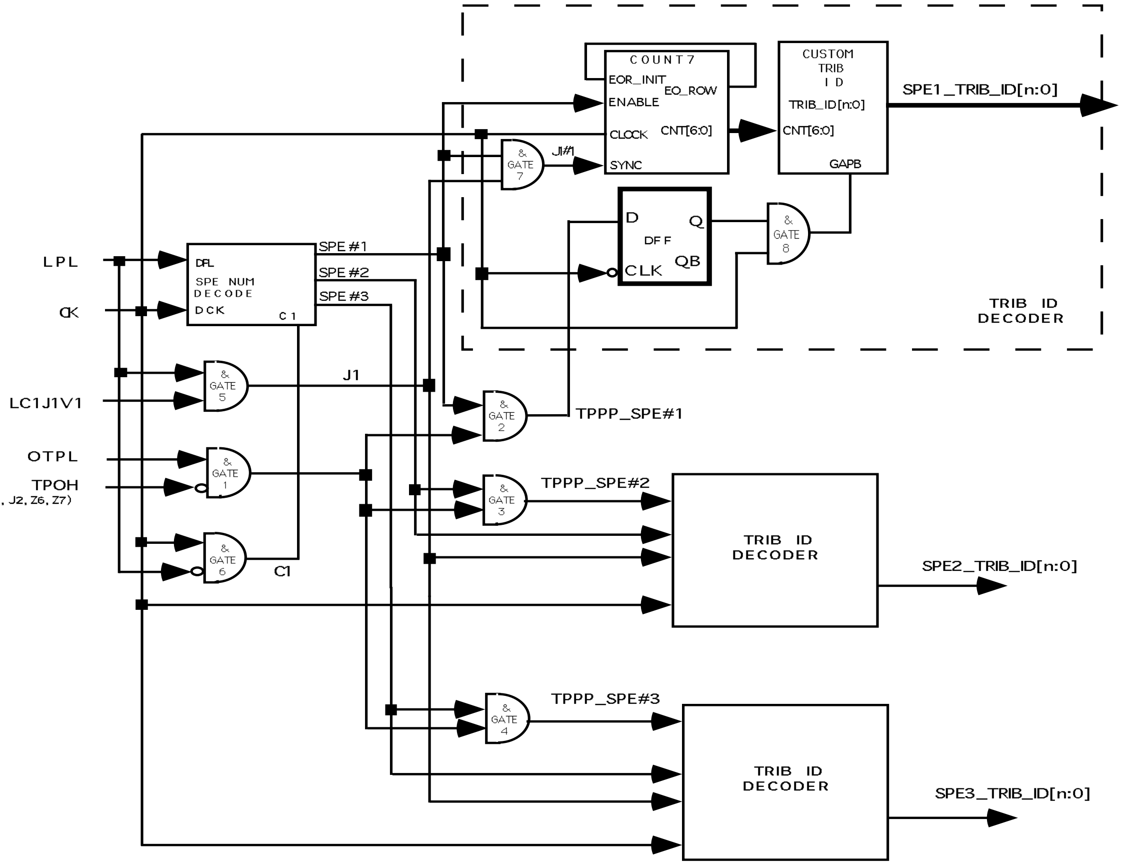


Figure 14: DROP Traffic Manager For The TUPP to HDLC Controller Interface

Note, for AU4 frame structures, the three TRIB ID DECODER blocks can be collapsed to a single TRIB ID BLOCK (since there is only one J1 byte) and the COUNT7 counter should be changed to a counter capable of counting 260 columns instead of the 87 columns capability shown in figure 14. Also, the SPE NUM DECODE block must be deleted since there is no longer a need to identify separate SPE's, and the output SPE#1 simply becomes LPL. The AND gates number 2, 3, 4 and 6 also become redundant. The same applies when the TUPP-PLUS is operating in locked mode with AU3 mapped SDH or STS-3 SONET, since the J1 bytes can all be located to be adjacent to each other.

SPE NUM DECODER
This block identifies the STS-1#1, STS-1#2 and STS-1#3 payload bytes and is identical to the block shown in figure 12.

ADD Traffic Manager (Tributary Level)

This block is in essence very similar to the functionality of the DROP Traffic Manager. However, for the case where the PPP HDLC controller is the generator of the AC1J1V1 and APL signals, it implicitly knows where the PPP data bytes reside. i.e. the functionality of the ADD Traffic Manager is swallowed up by the HDLC Controller. In such a case the implementation of the ADD Traffic Manager simply consists of two wires, one to signal AC1J1V1 and the other to signal APL from the HDLC Controller to the SPTX.

In the case where the AC1J1V1 and APL signals are derived from, say, a delayed version of the DROP bus OTPL, TPOH, LC1J1V1 and LPL signals (from the TUPP-PLUS) or from some other mapping master other than the HDLC Controller then the functionality of the ADD Traffic Controller is identical to the functionality shown for the DROP Traffic Controller in figure 14 with minor differences as is shown in figure 15 below.

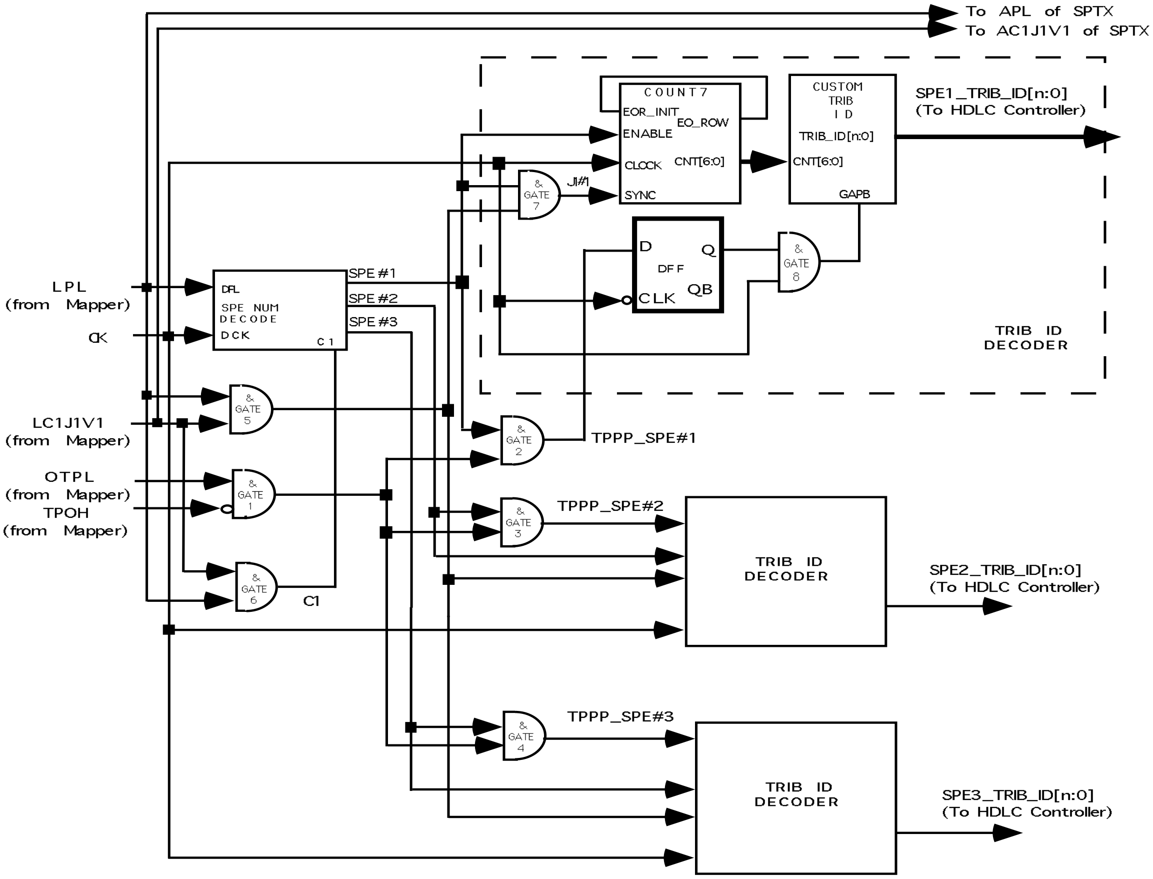


Figure 15: ADD Traffic Manager For The TUPP to HDLC Controller Interface

The SPE NUM DECODE block is the same as shown previously in figure 12.

DISCLAIMER

The circuits presented in this application note have not been built or simulated. These circuits are therefore preliminary in this release of this document.

REFERENCES

1) PMC-Sierra, Inc., PM5344 SPTX Data Sheet, Issue 4, Oct, 1995.

2) PMC-Sierra, Inc., PM5362 TUPP-PLUS Data Sheet, Issue 2, Apr, 1996.

3) PMC-Sierra,, Inc., PMC-950730 Path Overhead Processing And Payload Alignment Of An STS-12c/STM-4c Stream, Issue 1, July 27, 1995.

4) Network Working Group Request For Comments: RFC 1619 , May 1994.

NOTES

Seller will have no obligation or liability in respect of defects or damage caused by unauthorized use, mis-use, accident, external cause, installation error, or normal wear and tear. There are no warranties, representations or guarantees of any kind, either express or implied by law or custom, regarding the product or its performance, including those regarding quality, merchantability, fitness for purpose, condition, design, title, infringement of third-party rights, or conformance with sample. Seller shall not be responsible for any loss or damage of whatever nature resulting from the use of, or reliance upon, the information contained in this document. In no event will Seller be liable to Buyer or to any other party for loss of profits, loss of savings, or punitive, exemplary, incidental, consequential or special damages, even if Seller has knowledge of the possibility of such potential loss or damage and even if caused by Seller’s negligence.

PMC-960725 Issue date: July 14, 1996.

Printed in Canada