PMC-SIERRA

Transparent Generic Framing Procedure (GFP)

Technology White Paper

Steve Gorshe
Principal Engineer

Issue 1: May, 2002

PMC-2020811

Abstract

Generic Framing Procedure (GFP) is a new standard that has been developed to overcome inefficiencies and deficiencies in transporting data over existing ATM and Packet over SONET/SDH protocols. Transparent GFP (GFP-T) is an extension to GFP that has been developed to provide efficient, low-latency support for high-speed WAN applications including storage area networks (SANs). Rather than handling data on a frame-by-frame (or packet-by-packet) basis, transparent GFP-T handles the block-coded character streams (e.g., 8B/10B) directly. This paper describes the GFP protocol along with technical considerations and applications for transparent GFP.

About the Author

Steve Gorshe is a principal engineer in the Product Research Group and oversees ICs for SONET, optical transmission and access systems.

Currently Steve is a senior member of the IEEE and co-editor for the IEEE Communications magazine's *Broadband Access Series*. He is the chief editor for the ANSI T1X1 Subcommittee and is responsible for SONET and optical network interface standards. He has also been a technical editor for T1.105, T1.105.01, T1.105.02, and T1.105.07 within the SONET standard series as well as the ITU-T SG15 G.7041 (GFP) recommendation. He has 24 patents issued or pending and several published papers.

Revision History

Issue No.	Issue Date	Details of Change
1	May, 2002	Document created

Contents

List of Figures

List of Tables

1 Introduction

Several important high-speed LAN protocols use a Layer 1 block code in order to communicate both data and control information. The most common block code is the 8B/10B line code used for Gigabit Ethernet, ESCON, SBCON, Fiber Channel, FICON, and Infiniband. These have become increasingly important with the growing popularity of Storage Area Networks (SANs). The 8B/10B line code maps the 2^8 (256) possible data values into the 2^{10} (1024) values of the 10-bit code space. The code assignment is done such that the running number of ones and zeros transmitted on the line (the running disparity) remains balanced. Twelve of the 10 bit codes are reserved for use as control codes that may be used by the data source to signal control information to the data sink. In order to efficiently transport protocols using the 8B/10B line code through a public transport network such as SONET/SDH or the Optical Transport Network (OTN), it is necessary to carry both the data and the 8B/10B control code information. However, using the 8B/10B coding expands the data bandwidth by 25%, which is undesirable in the transport network. Among the alternative protocols for carrying these LAN signals through the SONET/SDH and OTN networks, the newly developed Generic Framing Procedure (GFP) [1] provides some advantages over ATM and Packet over SONET/SDH (POS). ATM requires a more complex adaptation process than GFP. POS requires terminating the client signal's Layer 2 protocol and re-mapping the signal into the Point-to-Point Protocol (PPP) over HDLC. This mapping further requires "escaping" the HDLC control characters found in the data stream, resulting in a non-deterministic bandwidth expansion.

GFP standardization began in the ANSI accredited T1X1 subcommittee, which chose to work with the ITU-T for the final version of the standard that has been published by the ITU-T [1]. The transparent version of GFP has been optimized for transparently carrying block-coded client signals with a minimum of latency. This paper describes the transparent GFP protocol along with some of the motivations and target applications that led to its development.

2 Transparent GFP Description

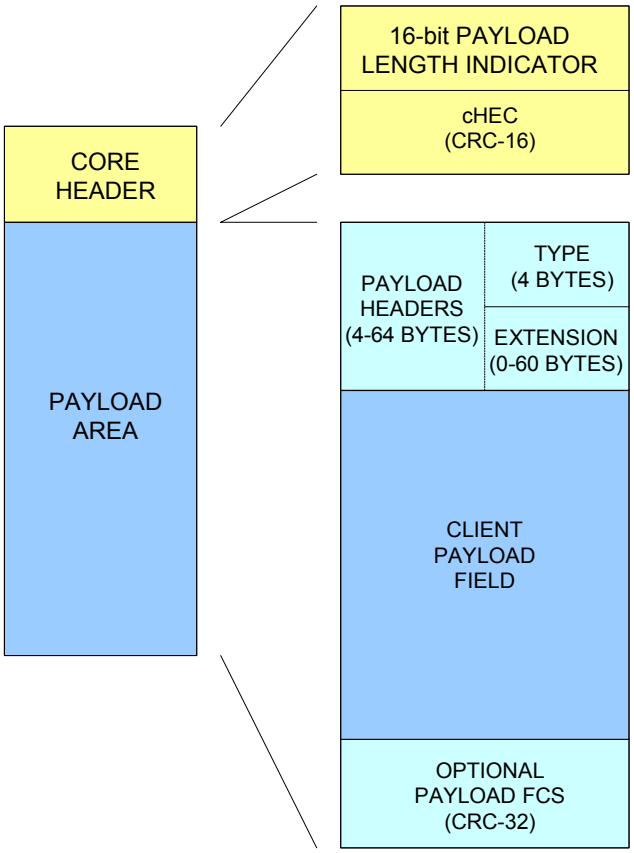
2.1 General GFP overview

The basic GFP frame structure is shown in Figure 1. The GFP core header consists of a two-octet length field, which specifies the length of the GFP frame's payload area in octets, and a CRC-16 error check code over this length field. To synchronize alignment to this frame structure, the framer looks for a 32-bit pattern that has the proper zero CRC remainder. It then confirms that this is the correct frame alignment by verifying that another valid 32-bit sequence exists immediately following where the length field specifies that the current frame ends.

In contrast, protocols such as HDLC rely on specific data patterns for frame delineation or to provide control information. These specific data patterns are not allowed in the payload as they would mimic the reserved characters and interfere with the proper operation of the protocol. The HDLC protocol requires additional escape bits or characters adjacent to the payload strings or bytes that mimic the reserved characters, which increases the amount bandwidth needed to convey the payload. A big drawback with this scheme is that the bandwidth expansion is non-deterministic. If the client payload data consists entirely of data emulating these reserved characters, then byte-stuffed HDLC protocols like POS will require nearly twice the bandwidth to transmit the packet than if the payload did not contain such characters. GFP avoids this problem by using only the information in its core header for frame delineation. Since no special characters are used for framing, there are no forbidden payload values that require escape characters. As well, the CRC-16 also provides a level of robustness by allowing single bit error to be corrected in the core header once frame alignment has been acquired.

In frame-mapped GFP (GFP-F), a single client data frame (e.g., an IP packet or Ethernet MAC frame) is mapped into a single GFP frame. The payload length is, therefore, variable for frame-mapped GFP. Furthermore, the client frame must be buffered in its entirety in order to determine its length. For transparent GFP, however, a fixed number of client characters are mapped into a GFP frame of pre-determined length. This makes the payload length static for transparent GFP. As its name implies, transparent GFP supports the transparent transport of 8B/10B control characters as well as data characters. This is a primary advantage of transparent GFP over frame-mapped GFP. In addition, frame-mapped GFP incurs the latency associated with buffering an entire client data frame, while transparent GFP requires only a few bytes of mapper/demapper latency. This lower latency is critical for SAN protocols, which are very sensitive to transmission delay.

Figure 1 GFP frame format



2.2 Transparent GFP 64B/65B Block Coding

The client 8B/10B codes are decoded into control codes and 8-bit data values. Eight of these decoded characters are then mapped into the eight payload bytes of a 64B/65B code. The leading (flag) bit of the 64B/65B code indicates whether there are any control codes present in that 64B/65B code. The presence of a control code is indicated by flag=1. The 64B/65B block structure for the various combinations of control codes and payload bytes is illustrated in Table 1. A control code byte consists of three fields. The first field consists of a single bit to indicate whether this byte contains the last control code in this 64B/65B block; it is set to logic 0 if it is the last. The next field is a 3-bit address (aaa --- hhh) indicating the original location of that control code in the client data stream relative to the other characters mapped into that 64B/65B block. The last field is a 4-bit code (C*n*) representing the control code. Since there are only twelve defined 8B/10B control codes, four bits are adequate to represent them. The control codes are placed in the leading bytes of the 64B/65B block, followed by the data bytes. Figure 2 illustrates an example mapping of 2 control and 6 data octets in the 64B/65B block.

Table 1 64B/65B Block Code Structure

Input Client Characters	Flag Bit	64-Bit (8-Octet) Field							
		Octet 0	Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7
All data	0	D1	D2	D3	D4	D5	D6	D7	D8
7 data, 1 control	1	0 aaa C1	D1	D2	D3	D4	D5	D6	D7
6 data, 2 control	1	1 aaa C1	0 bbb C2	D1	D2	D3	D4	D5	D6
5 data, 3 control	1	1 aaa C1	1 bbb C2	0 ccc C3	D1	D2	D3	D4	D5
4 data, 4 control	1	1 aaa C1	1 bbb C2	1 ccc C3	0 ddd C4	D1	D2	D3	D4
3 data, 5 control	1	1 aaa C1	1 bbb C2	1 ccc C3	1 ddd C4	0 eee C5	D1	D2	D3
2 data, 6 control	1	1 aaa C1	1 bbb C2	1 ccc C3	1 ddd C4	1 eee C5	0 fff C6	D1	D2
1 data, 7 control	1	1 aaa C1	1 bbb C2	1 ccc C3	1 ddd C4	1 eee C5	1 fff C6	0 ggg C7	D1
8 control	1	1 aaa C1	1 bbb C2	1 ccc C3	1 ddd C4	1 eee C5	1 fff C6	1 ggg C7	0 hhh C8

Legend:

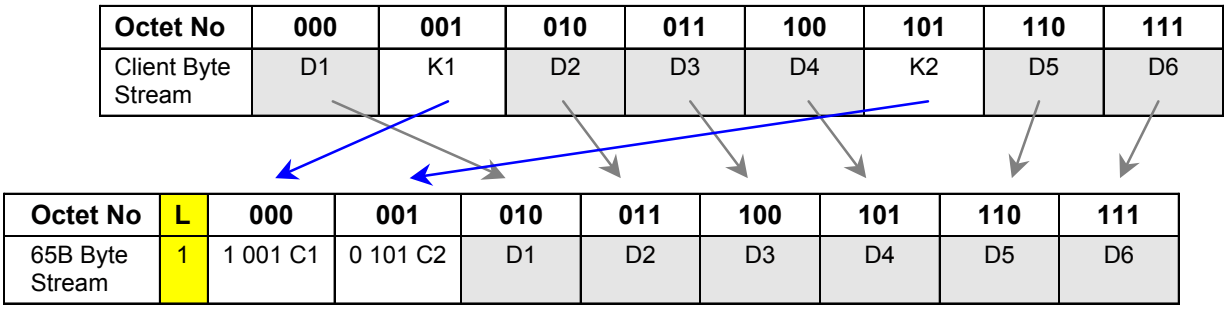
- Leading bit in a control octet (LCC) is logic 1 if there are more control octets and is logic 0 if this payload octet contains the last control octet in that block
- aaa = 3-bit representation of the 1st control code's original position (1st Control Code Locator)
- bbb = 3-bit representation of the 2nd control code's original position (2nd Control Code Locator)

…

- hhh = 3-bit representation of the 8th control code's original position (8th Control Code Locator)

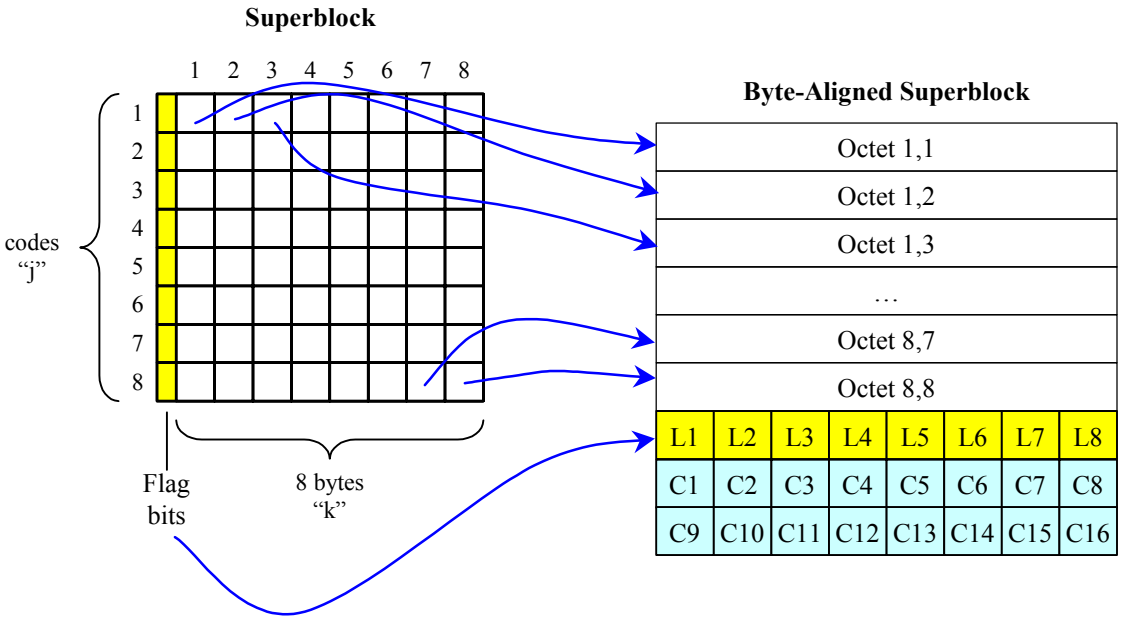
- Ci = 4-bit representation of the i^{th} control code (Control Code Indicator)
- Di = 8-bit representation of the i^{th} data value in order of transmission

Figure 2 Example of mapping Client Byte stream into 64B/65B block



Aligning the 64B/65B payload bytes with the SONET/SDH/OTN payload bytes simplifies parallel data path implementations as well as increases the data observability. This is accomplished by combining a group of eight 64B/65B codes into a superblock. The superblock structure, as shown in Figure 3, concatenates the payload bytes in order, then takes the leading flag bits of the eight constituent 64B/65B codes and groups them into a trailing byte. This is followed by a CRC-16, calculated over the bits of that superblock. The CRC-16 is discussed further in section 2.4.

Figure 3 Superblock structure for mapping 64B/65B code components into the GFP frame



where: Octet j, k is the k^{th} octet of the j^{th} 64B/65B code in the superblock

Lj is the leading (Flag) bit j^{th} 64B/65B code in the superblock

Ci is the i^{th} error control bit

2.3 Transport bandwidth considerations

The transparent GFP channel sizes are chosen to accommodate the client data stream under worst-case clock tolerance conditions – that is, when the transport clock is running at the slowest rate of its tolerance range, and the client clock is running at the fastest rate of its tolerance range. In the case of a SONET/SDH transport channel, transparent GFP is carried over virtually concatenated signals. Multiple SONET Synchronous Payload Envelopes (SPEs) / SDH Virtual Containers (VCs) are grouped together to form a higher-bandwidth pipe between the endpoints of the virtually concatenated path. The constituent SPEs/VCs in the virtually concatenated path do not need to be time-slot contiguous. This greatly simplifies the provisioning and increases the flexibility of virtual concatenation. Furthermore, virtual concatenation is transparent to the intermediate nodes; only the end points of the virtually concatenated path need to be aware of its existence.

Virtually concatenated signals are indicated with the nomenclature <SPE/VC type>-*X*v, where *X* indicates the number of SPEs/VCs that are being concatenated. For example, STS-3c-7v is the virtual concatenation of seven STS-3c SPEs, which is equivalent to VC-4-7v for SDH. Virtual Concatenation is specified in ITU-T [7], ANSI [8] and ETSI [9]. Table 2 shows the minimum virtually concatenated channel size that can be used for various transparent GFP clients.

Table 2 Virtually concatenated channel sizes for various transparent GFP clients

Client Signal	Native (Unencoded) Client Signal Bandwidth	Minimum Virtually-Concatenated Transport Channel Size	Nominal Transport Channel Bandwidth	Minimum Number of Superblocks per GFP Frame	Worst/Best-case Residual Overhead Bandwidth[1]	Best case client management payload bandwidth[3]
ESCON	160 Mbit/s	STS-1-4v / VC-3-4v	193.536 Mbit/s	1	5.11 Mbit/s / 24.8 Mbit/s	6.76 Mbit/s
Fiber Channel	850 Mbit/s	STS-3c-6v / VC-4-6v	898.56 Mbit/s	13	412 Kbit/s / 85.82 Mbit/s	2.415 Mbit/s
Gbit Ethernet	1.0 Gbit/s	STS-3c-7v / VC-4-7v	1.04832 Gbit/s	95	281 Kbit/s / 1.138 Mbit/s	376.5 kbit/s
Infiniband[2]						

NOTES:

1) The worst-case residual bandwidth occurs when the minimum number of superblocks is used per GFP frame. The best case occurs for the value of N that allows exactly one client management frame per GFP data frame. A 160-bit client management frame was assumed for the best case (with a CRC-32). For both cases, it was assumed that no extension headers were used.

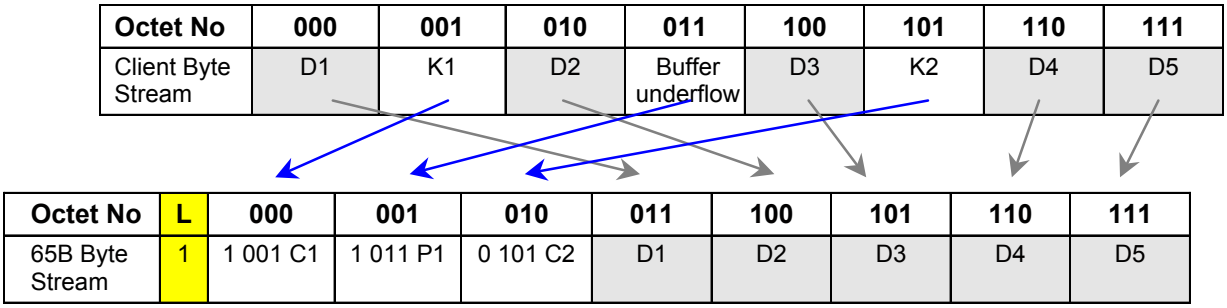
2) The requirements for the transport of Infiniband over transparent GFP have not yet been established.

3) The best-case client management payload bandwidth assumes 8 “payload” bytes per client management frame and the best-case residual overhead bandwidth conditions.

In practice, the SONET/SDH channel must be slightly larger in bandwidth than what is needed to carry the GFP signal. As a result the GFP mapper's client signal ingress buffer will underflow. There are two ways to handle this situation. One approach is to buffer an entire transparent GFP frame's worth of client data characters before starting to transmit that GFP frame. However, this approach increases the mapper's latency and buffer size. A second approach, which was adopted for the standard, is to use a dummy 64B/65B control code as a 65B_PAD character. Whenever there is no client character available in the ingress buffer, the mapper treats the situation the same as if a client control character was present and inserts the 4-bit 65B_PAD character. This is illustrated in Figure 4. The demapper at the other end of the GFP link recognizes this character as a dummy pad and removes it from the reconstituted data stream. By using the 65B_PAD character the mapper ingress buffer size is reduced to effectively eight bytes (the amount of data required to form a 64B/65B block) plus the number of bytes that can accumulate during the SONET/SDH overhead and the GFP frame overhead bytes. An eight-byte latency always exists since the mapper cannot complete the 64B/65B block coding until it knows whether there are any control codes present in the eight characters that will comprise that block.

As will be discussed in section 2.5, client management frames have been proposed for GFP that would use this "spare" bandwidth for client management applications. These client management frames would be up to 20 bytes in length (including the GFP encapsulation bytes) and, by having a lower priority than the client data, would only be sent when the ingress buffer is nearly empty. To support these client management frames an additional 20 bytes must be added to the ingress buffer requirements.

Figure 4 Example of an insertion of the 65B_PAD Character



2.4 Error control considerations

2.4.1 Error Detection

8B/10B codes have a built-in error detection capability where a single bit error will always result in an illegal code. But increasing bandwidth efficiency by remapping the data from 8B/10B codes into 64B/65B codes sacrifices much of this error detection capability. There are four situations where bit errors can cause significant problems with the 64B/65B codes. The first and most serious problem results when the code's leading flag bit is received in error. Since the value of this flag bit indicates whether the block contains control codes and data, or just data, an error here causes the bytes to be misinterpreted. For example, if the original block contained any control codes, these codes will be interpreted as data. If the original block contained only data, some of these data bytes may be interpreted as control codes. The number of data bytes that are erroneously interpreted as control codes depends on the value of the first bit[1] of the bytes and whether the values of the location address bit positions contain increasing values (which is always the case for a legal block). Data that is erroneously converted into control codes could truncate a client data frame, which in turn causes error detection problems for the client data since there is a possibility that the truncated client frame appears to have a correct CRC value. The second, similar problem occurs when a block correctly contains control characters and the last control code indicator bit is affected by an error. The third problem results when errors occur in the control code location address. This will cause it to be placed in the wrong sequence by the demapper. The fourth problem results when errors occur in the 4-bit control code value. This will cause the demapper to generate an incorrect control code. Any error that results in a spurious or incorrect control code can have potentially serious consequences.

To improve the error detection to address these potential problems a CRC-16 is added to each superblock. Once an error is detected, the most reliable mechanism for error control is for the demapper to discard all of the data in the errored superblock. For those clients that have defined such a code, the data is "discarded" by having the demapper output 10B_ERROR 8B/10B codes for all the characters in the superblock. For those clients that do not have error codes defined, the demapper can output another, illegal 8B/10B character for all of the characters in the superblock. Optionally, the CRC-16 can also allow the possibility of single error correction.

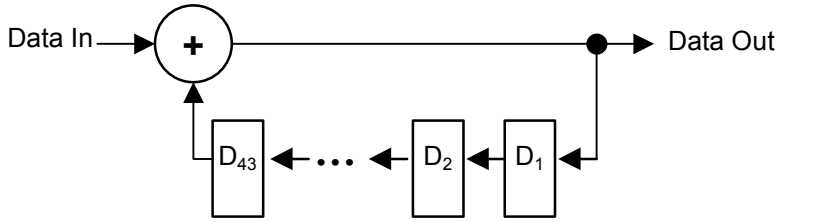
2.4.2 Implications of payload scrambling

The payload area of the GFP frame is scrambled with a self-synchronous scrambler to address both the physical properties of the transport medium and the desire for robustness in public networks. In SONET/SDH and OTN the data is passed through a SONET/SDH/OTN frame-synchronous scrambler and then transmitted using a non-return-to-zero (NRZ) line code. This NRZ line code controls the laser, turning it on for the bit period representing a "1" and turning it off to represent a "0." The NRZ line code is used because it is simple and bandwidth efficient. However, the disadvantage of NRZ is that the receiver clock and data recovery circuits can lose synchronization after a long string of either 0s or 1s. The frame-synchronized scrambler, which is reset at a regular interval based on the SONET/SDH/OTN frame, randomizes the user data to limit the length of these consecutive strings, and is adequate to defend against normally occurring

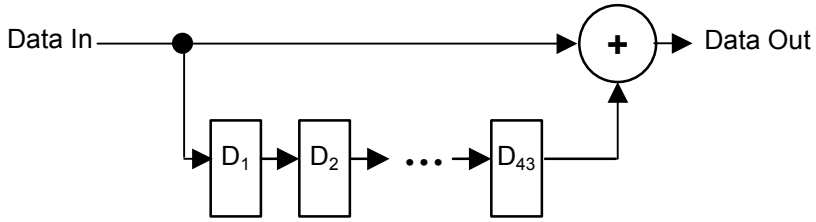
[1] recall that this bit position indicates the last control code

user data patterns. Unfortunately, since the user can send any data pattern, it is possible for a malicious user to choose a packet payload that is the same as the frame-synchronized scrambler sequence. If this sequence lines up in the correct position in the frame, it "undoes" the scrambling and results in an adequately long string of 0s or 1s that causes the receiver to lose synchronization. The network's reaction to the resulting loss of synchronization is to take down the link while the receiver attempts to recover, thus denying the link to other users for the duration of the recovery. This problem was originally discovered in ATM networks and is exacerbated by the longer frames used in POS or GFP. To address this danger, a self-synchronous scrambler was chosen for ATM, POS, and GFP with a scrambler polynomial of $x^{43}+1$. The scrambler takes each bit of the payload area and exclusively ORs it with the scrambler output bit that precedes it by 43 bit positions, as shown in Figure 5(a). The scrambler state is retained between successive GFP frames, making it much more difficult for a user to purposely choose a malicious payload pattern. The decoder's descrambler shown in Figure 5(b) reverses this process, restoring the user payload data.

Figure 5 Payload self-synchronous scrambler/descrambler



(a) x^{43} + 1 Scrambler



(b) x^{43} + 1 Descrambler

The same payload scrambling technique is used for both frame-mapped and transparent GFP, where all of the GFP payload bits including the superblock CRC bits are scrambled. As a result, the superblock CRC is calculated over the superblock payload bits *prior* to scrambling and is checked at the decoder *after* descrambling. Unfortunately, the drawback of self-synchronous scramblers is that each transmission error produces a pair of errors in the descrambled data. In the case of the $x^{43}+1$ scrambler polynomial, the errors are 43 bits apart. The superblock CRC must be able to cope with this error multiplication if it is to be of any use. Investigations have shown [2], [10] that a CRC will preserve its error detection capability in such a situation as long as the scambler polynomial and the CRC generator polynomial have no common factors. However, all of the standard CRC-16 polynomials contain x+1 as a factor, which is also a factor in the $x^{43}+1$

(or any x^n+1) scrambler polynomial. Therefore a new CRC generator polynomial is required that preserves the triple error detection capability (which is the maximum achievable over this block size) and does not have any common factors with the scrambler. Furthermore, in order to perform single error correction, the syndromes for single errors and for double errors spaced 43 bits apart must all be unique [10]. The CRC-16 polynomial $x^{16} + x^{15} + x^{12} + x^{10} + x^4 + x^3 + x^2 + x + 1$ selected for the Transparent GFP superblock has both these desired properties, and hence retains its triple error detection and optional single error correction capabilities in the presence of the scrambler [3], [4], [10].

2.5 Transparent GFP Client Management Frames

As was previously mentioned, there is some residual "spare" bandwidth in the SONET/SDH channel for each of the client signal mappings. Table 2 shows the amount of this "spare" bandwidth, which depends on the efficiency of the mapping, which in turn is partially a function of the number of superblocks used in each GFP frame. The residual bandwidth can be used as a client management overhead channel for client management functions, as described in section 2.5.2 and [5]. Client Management Frames (CMFs) can also be used for downstream indication of Client Signal fail, as described in 2.5.1.

CMFs have the same structure as GFP client data frames but are identified by the payload type code PTI = 100. Like GFP client data frames CMFs have a Core header and a Payload type header (both with two-byte HEC), and an optional 32-bit FCS. The total CMF payload size in transparent mapped mode is recommended to be no greater than eight bytes.

In applications where the CMF payload is eight bytes, no FCS is used and no Extension header is used (which would add a total of 20 bytes) the payload efficiency is 50%. Using the FCS and especially using the Extension headers will greatly reduce the efficiency of the client management frames.

2.5.1 Client Signal Fail Indication

GFP uses Client Management Frames to indicate Client Signal Fail (CSF) to the far-end GFP equipment. When a failure defect is detected in the ingress client signal a GFP CMF is transmitted immediately after the current frame. This CMF has the Payload Type Header set to PTI = 100, PFI = 0 (no FCS), an appropriate EXI field, and the UPI = 0000 0001 (for Loss of Client Signal) or UPI=0000 0010 (for Loss of Client Character synchronization). Since a CSF can occur in the middle of a GFP client data frame and the payload length indication has already been transmitted at the beginning of the client data frame, the remainder of the current frame will be filled with 10B_ERR codes to give it the required length. Then the CSF CMF can be sent.

Client Management Frames indicating CSF are sent on a regular interval, every 100 ms<T<1000 ms in order to prevent:

- overwhelming the receiver with frequent CSF indications, and
- excessively hogging the bandwidth with CSF indications for just one channel in the case of a frame multiplexed scenario.

When the GFP client sink receives the CSF indication, the receiver declares a sink client signal failure and outputs either 10B_ERROR or another illegal 8B/10B code on the client egress signal. If the CSF condition is a Loss of Signal and lasts for an extended time, the client egress output signal transmitter (e.g., the laser) can be turned off for protection. The CSF condition at the receiver is cleared when a valid client data frame is received, or when less than *N* CSF indications in *N* X 1000ms is received. A value of 3 is suggested for *N*.

2.5.2 CMF potential uses

Far-end Performance Reporting

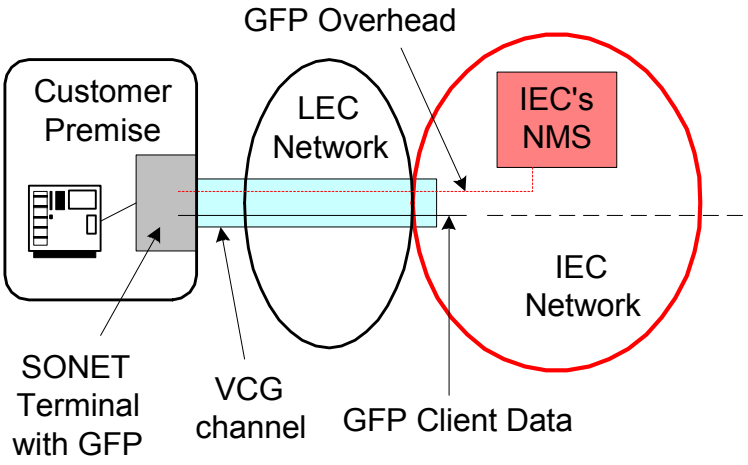
The most universal Client Management Frame application is reporting client-specific performance information from the far-end of the GFP link. The GFP receiver can report performance statistics such as the bit error rate (BER) or the ratio of good to bad client frames either on a periodic basis or when queried. Far-end client-specific performance reporting allows both ends of the GFP link to see the status of both directions of the GFP link, which is valuable when one of the ends is in an unmanned office or when the link crosses carrier domains.

Remote Management

If both ends of the GFP link are owned by the same carrier and the intervening SONET/ SDH/ OTN network is owned by another operator, then the opportunity exists to send provisioning commands using CMFs. It is common for interexchange carriers (IECs) to provide the customer premise terminal equipment (CPE) and rely on a local exchange carrier (LEC) to provide the connection between this CPE and the interexchange network (see Figure 6). Ideally, the IEC would like to manage the CPE as part of its own network, freeing the customer from having to manage the equipment, and potentially receiving revenue by providing this management service. Normally, management information is communicated through a SONET/SDH Section Data Communication Channel (SDCC). However, carriers do not allow SDCC data to cross the network interfaces into their networks, preventing unwanted control access. This also prevents the IEC from exchanging management communications with the CPE through the intervening LEC network. The transparent GFP Client Management Frames provide a mechanism to tunnel the SDCC information through the intervening network.

The last column of Table 2 shows the maximum amount of “payload” capacity can be derived from the client management frames for SDCC tunneling or for other OAM applications. With the assumptions stated in the table notes and assuming a 20-byte client management frame with an 8-byte payload field, there is adequate bandwidth available to carry a 192 kbit/s SDCC channel for all client signal types.

Figure 6 SDCC tunneling application example with transparent GFP



3 Potential Extensions to Transparent GFP

There are three main areas where extensions are possible to GFP-T:

1) To support other client signal types – ETSI standard Digital Video Broadcast has recently been proposed as a GFP-T mapping. Infiniband is another potential LAN signal that could be mapped into GFP-T. If an application arises for it, a recent proposal has shown that it would also be possible to map the 4B/5B 100Base Ethernet signal into GFP-T in a manner similar to the 8B/10B coded mappings [6].

2) To support other transport media – Another potential extension is the direct mapping of GFP onto a wavelength in an OTN network (i.e., with no underlying SONET/SDH or OTN transport signal). This extension would require the definition of a GFP physical layer.

3) To support other client management applications – Two applications have already been described in the preceding section that make use of the versatile CMFs. Other extensions could include using the CMFs as a trace function to guarantee correct connectivity of the GFP-T stream as it passes through a network element that routes GFP signals.

4 Conclusions

Transparent GFP provides an efficient mechanism for mapping constant bit rate block coded data signals across a SONET/SDH or OTN network. By performing the mapping on a client character basis rather than a client frame basis the transport latency is significantly reduced. Reducing latency is a critical issue for SAN protocols including Gigabit Ethernet. Translating the client block codes into the more efficient 64B/65B mapping provides a significant bandwidth efficiency increase, while the superblock structure itself provides robust error performance. Transparent GFP also improves the performance monitoring capability for the transport layer, while the ability to tunnel SDCC management information through an intervening network provides a powerful extension to network providers' capabilities.

5 References

[1] ITU-T Recommendation G.7041/Y.1303 Generic Framing Procedure -, S. Gorshe - technical editor.

[2] T1X1.5/2001-094, "Impact of $x^{43} + 1$ Scrambler on the Error Detection Capabilities of Ethernet CRC," standards contribution from N. Figueira, Nortel Networks, March 2001.

[3] T1X1.5/2001-125, "Recommended CRC-16 Polynomial for the Transparent GFP Superblock and Associated New Text," standards contribution from S. Gorshe, PMC-Sierra, June 2001

[4] T1X1.5/2001-174, "Optimum CRC-16 Polynomial for the Transparent GFP Superblock," standard contribution from S. Gorshe, PMC-Sierra, Sept. 2001

[5] T1X1.5/2001-148, "Proposed Draft Text for GFP OAM Frames," standards contribution from T. Wilson (Nortel Networks), M. Scholten (AMCC), and S. Gorshe (PMC-Sierra), June 2001.

[6] T1X1.5/2001-177, "Proposed ITU-T Contribution for Adding a 4B/5B Ethernet Mapping to Transport GFP," standards contribution from P. Thaler (Agilent) and S. Gorshe (PMC-Sierra), Sept. 2001

[7] ITU-T Recommendation G.707/Y.1322 Network node interface for the Synchronous Digital Hierarchy (SDH)

[8] ANSI standard T1.105 Synchronous Optical Network (SONET) -Basic Description including Multiplex Structure, Rates and Formats

[9] ETSI standard EN 300 417-9-1 Synchronous Digital Hierarchy (SDH) concatenated path layer functions

[10] S. Gorshe, "CRC-16 Polynomials Optimized for Applications Using Self-Synchronous Scramblers," paper accepted for publication in *Proc. of ICC2002.*

Head Office:
PMC-Sierra, Inc.
#105 - 8555 Baxter Place
Burnaby, B.C. V5A 4V7
Canada
Tel: 604.415.6000
Fax: 604.415.6200

To order documentation, send email to:
document@pmc-sierra.com
or contact the head office,
Attn: Document Coordinator

All product documentation is available on our web site at:
http://www.pmc-sierra.com
For corporate information, send email to:
info@pmc-sierra.com

PMC-2020811 (1)