Transparent Generic Framing Procedure (GFP): A Protocol for Efficient Transport of Block-Coded Data through SONET/SDH Networks

Steven S. Gorshe, PMC-Sierra
Trevor Wilson, Nortel Networks

ABSTRACT

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

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, Fibre Channel, FICON, and Infiniband, which have become increasingly important with the growing popularity of storage area networks (SANs). Since both client data bytes and data source to sink control information are encoded into the 8B/10B codes, efficient transport of these protocols through a public transport network such as synchronous optical network/synchronous digital hierarchy (SONET/SDH) or the optical transport network (OTN) requires transporting both the data and the 8B/10B control code information. 8B/10B coding, however, adds a 25 percent data bandwidth expansion that is undesirable in the transport network.

The previously available protocols for LAN transport through SONET/SDH networks were asynchronous tranfer mode (ATM) and packet over SONET/SDH (POS). ATM is relatively inefficient from a bandwidth utilization standpoint and typically requires a much more complex adaptation process than GFP. POS requires terminating the client signal's layer 2 protocol and remapping the signal into Point-to-Point Protocol (PPP) over HDLC, which suffers from a nondeterministic bandwidth expansion, discussed in the section on bandwidth considerations. Also, neither ATM nor POS support the transparent transport of the 8B/10B control characters. In order to overcome the shortcomings of ATM and POS, GFP standardization began in the American National Standards Institute (ANSI) accredited T1X1 subcommittee, which chose to work with the International Telecommunication Union — Telecommunication Standardization Sector (ITU-T) on the final version of the standard, which has been published by the ITU-T [1]. The transparent version of GFP (GFP-T) has been optimized for transparently carrying block-coded client signals (i.e., both the data and the 8B/10B control codes) with minimal latency. This article begins with a description of the transparent GFP protocol followed by some special considerations such as bandwidth, error control, and client management. Potential extensions to the transparent GFP protocol are then also briefly discussed.

0163-6804/02/$17.00 © 2002 IEEE

TRANSPARENT GFP DESCRIPTION

GENERAL GFP OVERVIEW

The basic GFP frame structure is shown in Fig. 1. Protocols such as HDLC that rely on specific data patterns for frame delimiting or control information require a nondeterministic amount of bandwidth due to the need for additional escape bits or characters adjacent to the payload strings or bytes that mimic these reserved characters. The amount of expansion is thus data-pattern-dependent. In the extreme case, if the client payload data consists entirely of data emulating these reserved characters, byte-stuffed HDLC protocols like POS require nearly twice the bandwidth to transmit the packet than if the payload did not contain such characters. GFP avoids this problem by using information in its core header for frame delimitation. Specifically, the GFP core header consists of a two-octet-long field that specifies the length of the GFP frame's payload area in octets, and a cyclic redundancy check (CRC-16) error check code over this length field. The framer looks for a 32-bit pattern that has the proper zero CRC remainder and then confirms that this is the correct frame alignment by verifying that another valid 32-bit sequence exists immediately following where the length field specified the current frame ends. Since no special characters are used for framing, there are no forbidden payload values that require escape characters. (Note that CRC-16 also provides robustness by allowing single error correction on 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 medium access control, MAC, frame) is mapped into a single GFP frame. For transparent GFP, however, a fixed number of client characters are mapped into a GFP frame of predetermined length. Hence, the payload length is typically variable for frame-mapped GFP and static for transparent GFP. One of the primary advantages of GFP-T over GFP-F is that GFP-T supports the transparent transport of 8B/10B control characters as well as data characters. In addition, GFP-F typically incurs the latency associated with buffering an entire client data frame at the ingress to the GFP mapper. As discussed below, GFP-T requires only a few bytes of mapper/demapper latency. This lower latency is a critical issue for SAN protocols, which are very sensitive to transmission delay. (Note that GFP-F is best suited to applications where latency is less important than bandwidth efficiency. For example, if the client signal is lightly loaded, GFP-F allows mapping the packets into a smaller transport channel or potentially frame-multiplexing them into a shared channel with GFP frames from other client signals. Alternatively, GFP-F could make use of the Link Capacity Adjustment Scheme [G.7042] for handling client signals that experience temporary changes to their required bandwidth.)

TRANSPARENT GFP 64B/65B BLOCK CODING

The 8B/10B line code maps the $2^8 = 256$ possible data values into the $2^{10} = 1024$ value 10-bit code space such that the running number of ones and zeros transmitted on the line (the running disparity) remains balanced over very short intervals. 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. The first step of GFP-T encoding in the source adaptation process is to decode the client 8B/10B codes 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 (with flag = 1 indicating the presence of a control code). The 64B/65B block structure for various numbers of control codes is illustrated in Fig. 2. Control codes are placed in the leading bytes of the 64B/65B block as illustrated in Fig. 2. A control code byte consists of a bit to indicate whether this byte contains the last control code in that 64B/65B block (= 0 if it is the last), 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, and a 4-bit code (C_n) representing the control code. Since there are only 12 defined 8B/10B control codes, 4 bits are adequate to represent them. One of the remaining 4-bit codes is used to communicate that an illegal 8B/10B character has been received by the GFP source adaptation process so that the GFP receiver can output an equivalent illegal 8B/10B character to the client signal sink. Figure 3 illustrates mapping of control and data octets in the 64B/65B block.

Aligning the 64B/65B payload bytes with the SONET/SDH/OTN payload bytes simplifies parallel data path implementations as well as increasing the payload data observability within

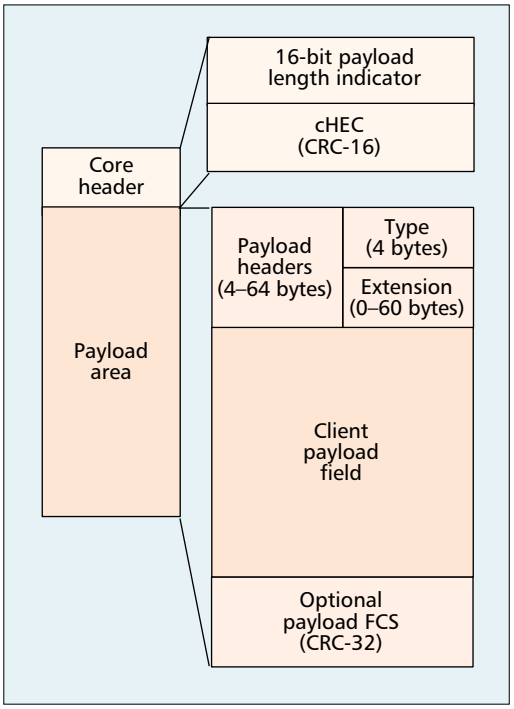

Figure 1. *GFP frame format.*

Aligning the 64B/65B payload bytes with the SONET/SDH/OTN payload bytes simplifies parallel data path implementations and increases the payload data observability within the SONET/SDH stream.

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 data	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) = 1 if there are more control octets and = 0 if this payload octet contains the last control octet in that block
- aaa = 3-bit representation of the first control code's original position (first control code locator)
- bbb = 3-bit representation of the second control code's original position (second control code locator)

. . .

- hhh = 3-bit representation of the eighth control code's original position (eighth control code locator)

- C*i* = 4-bit representation of the *i*th control code (control code indicator)
- D*i* = 8-bit representation of the *i*th data value in order of trnsmission

Figure 2. *64B/65B block code structure.*

the SONET/SDH stream. In order to achieve this alignment, a group of eight 64B/65B codes are combined into a superblock. The superblock structure, as shown in Fig. 4, takes the leading flag bits of the eight constituent 64B/65B codes and groups them into a trailing byte followed by a CRC-16 over the bits of that superblock. CRC-16 is discussed further in the section on error control considerations.

Transport Bandwidth Considerations

GFP-T channel sizes are chosen to accommodate the client data stream under worst case clock tolerance conditions (i.e., for the slowest end of the transport clock and fastest end of the client clock tolerance). In the case of SONET/SDH, while GFP-T can be carried over contiguously concatenated channels, it will typically be carried over virtually concatenated signals. The concept of virtual concatenation is one in which 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. Note that the constituent SPEs/VCs do not need to be time-slot-contiguous, which greatly simplifies the provisioning and increases the flexibility of virtual concatenation. Another advantage of virtual concatenation is that it is transparent to intermediate nodes with only the endpoints of the virtually concatenated path needing to be aware of its existence. The nomenclature for indicating a virtually concatenated signal is <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 [2], ANSI [3], and the European Telecommunications Standards Institute (ETSI) [4]. Table 1 shows the minimum virtually concatenated channel size that can be used for various GFP-T clients.

In practice, the SONET/SDH channel must be slightly larger than that needed to carry the GFP signal, a consequence of which is that 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 GFP-T frame's worth of client data characters prior to beginning the transmission of that

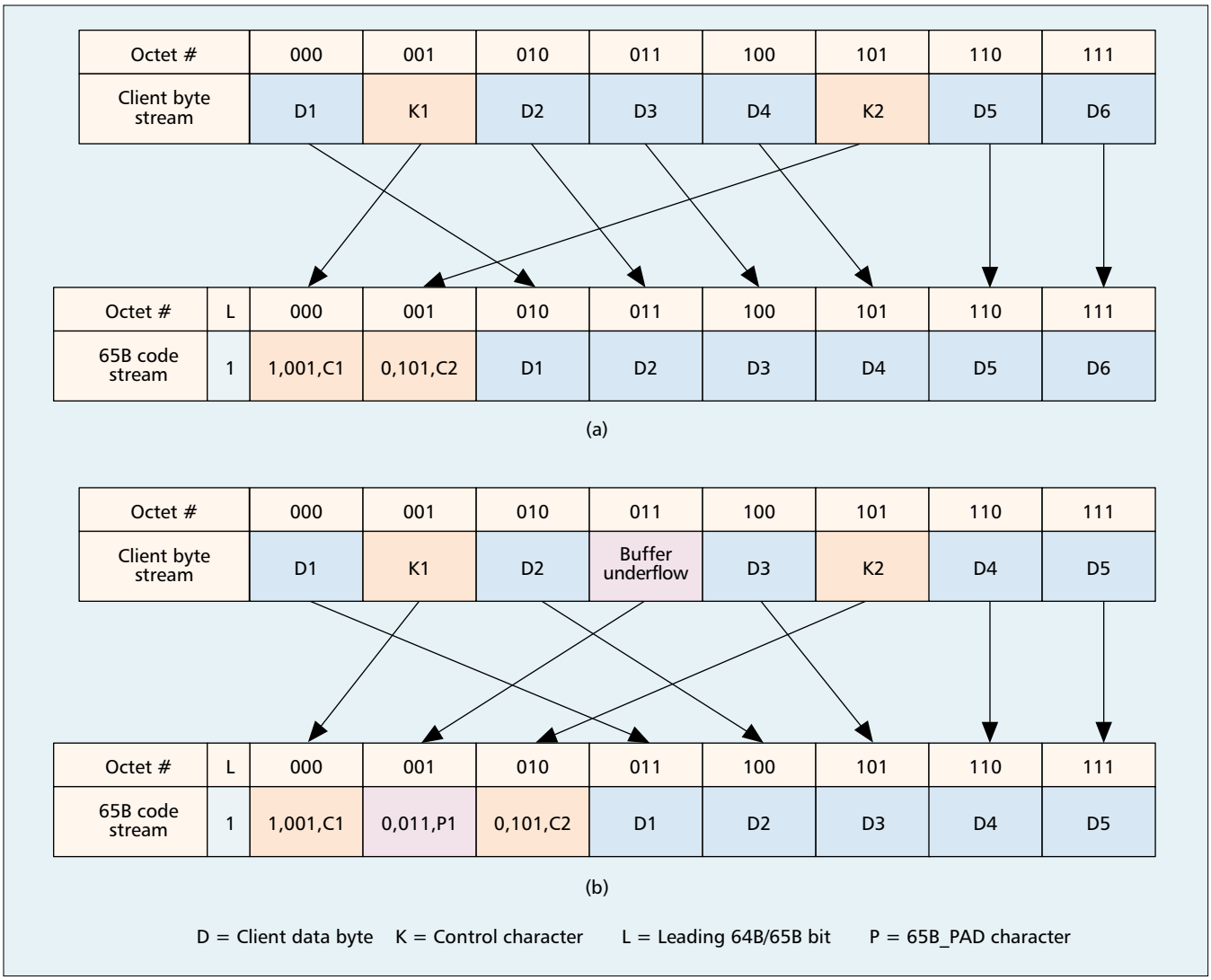

Figure 3. *Examples of mapping a client byte stream into a 64B/65B block: a) with control and data bytes; b) including 65B_PAD insertion.*

GFP frame. This approach would increase the mapper 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 will treat the situation the same as if a client control character was present and will insert the 4-bit 65B_PAD character. Figure 3b illustrates the insertion of a 65B_PAD character. The demapper at the other end of the GFP link recognizes this character as a dummy pad and removes it from the data stream. The result of using this 65B_PAD character is that the mapper ingress buffer size is reduced to effectively eight bytes (i.e., 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 is always required 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 make up that block.

As discussed below, client management frames (CMFs) have been proposed for GFP that would make use of this "spare" bandwidth for client management applications. These CMFs would be up to 20 bytes long (including GFP encapsulation bytes) and, due to having lower priority than the client data, would only be allowed to be sent when the ingress buffer is nearly empty. Support for these CMFs adds 20 bytes to the ingress buffer requirements to accommodate the data arriving during the transmission of a CMF.

Demapping of the GFP-T signal entails the removal of the 65B_PAD characters, and removal and interpretation of the interframe CMFs when present. Assuming that the egress of the client signal is done using a constant rate local clock, if the egress buffer becomes empty due to reception of 65B_PAD characters and/or CMFs, interpacket fill words must be inserted according to the client signal type rules.

Error Control Considerations

8B/10B codes have built-in error detection capability since a single bit error will always result in an illegal code. The increased bandwidth efficiency gained by decoding the 8B/10B codes and

The reasons for using a self-synchronized payload scrambling process are related to the physical properties of the transport medium and the desire for robustness in public networks.

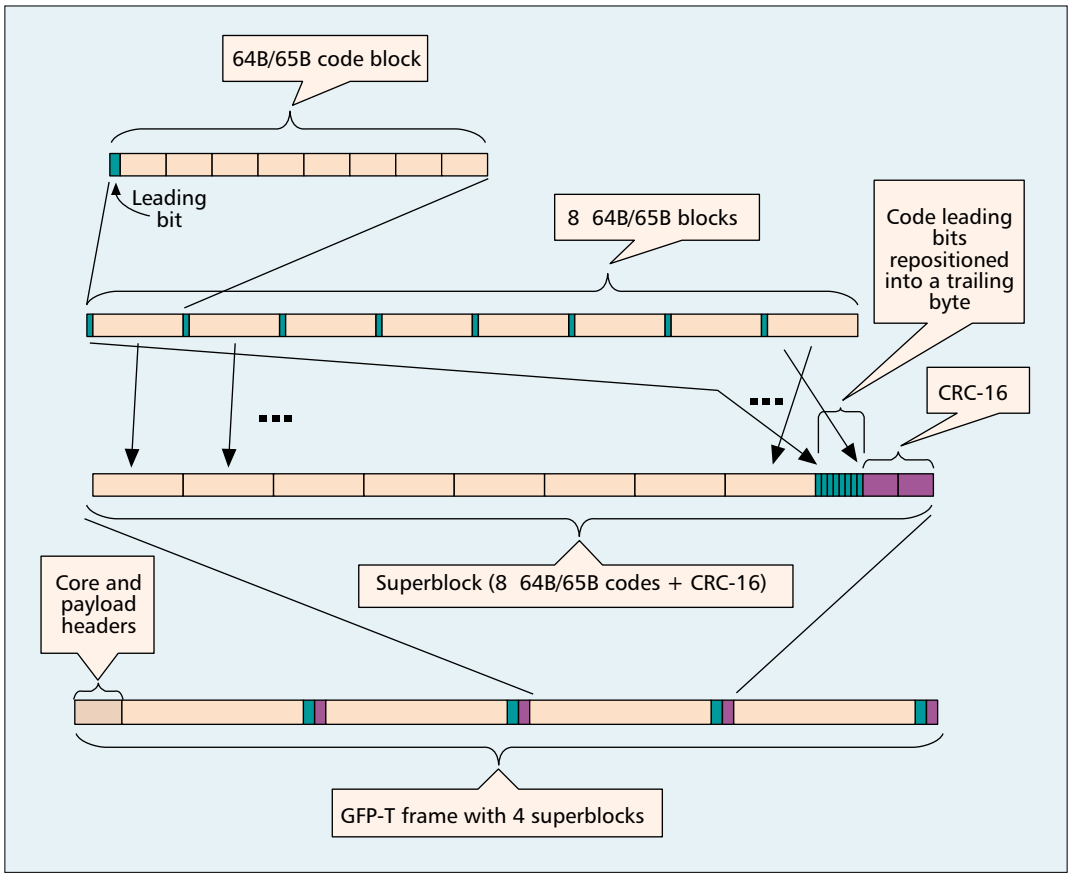

Figure 4. *Superblock construction for mapping 64B/65B code components into the GFP frame.*

remapping the data into 64B/65B codes comes at the expense of much of this error detection capability. There are four situations in which errors can cause significant problems with 64B/65B codes. The first and most serious problem results if the leading flag bit of the 64B/65B code is received in error. If the original block contained control codes, these codes will be interpreted as data, and if the original block contained only data, some of these 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 (i.e., the last control code indicator bit position) of the bytes and whether the values of the location address bit positions contain increasing values (which would always be the case for a legal block). Data erroneously converted into control codes could cause the truncation of a client data frame, which in turn can cause error detection problems for the client data since there is a possibility of the truncated client data frame appearing to have a correct CRC value. A similar situation occurs when control characters are present and the last control code indicator bit is affected by an error. Also, errors in the control code location address will cause it to be placed in the wrong sequence by the demapper, and errors in a 4-bit control code value will cause the demapper to generate an incorrect control code. Any error that results in a spurious or incorrect control code has potentially serious consequences.

It is these potential error problems that lead to the addition of a CRC-16 to each superblock. The most reliable mechanism for error control is for the demapper to discard all of the data in a superblock in which an error is detected. The data is discarded by having the demapper output 10B_ERROR 8B/10B codes for those clients that have defined such a code, or another illegal 8B/10B character for all of the characters in that superblock. Note that, as discussed below, the CRC-16 optionally allows the possibility of single error correction.

The payload area of the GFP frame is scrambled with a self-synchronous scrambler, and another error control issue concerns the interaction between the GFP payload scrambler and the superblock CRC-16. To understand the issue here, it is helpful to first understand the rationale and implementation behind the payload scrambler.

The reasons for using a self-synchronized payload scrambling process are related to the physical properties of the transport medium and the desire for robustness in public networks. The line code used for SONET/SDH and OTN is non-return-to-zero (NRZ) (after the data has been passed through a SONET/SDH/OTN frame-synchronous scrambler). For NRZ, the laser is turned on for the bit period to represent a 1 and off to represent a 0. The advantage of the NRZ line code is its simplicity and bandwidth efficiency. The disadvantage of NRZ, however, is that the receiver clock and data recovery circuits can lose synchronization after a long string of either 0s or 1s. The frame-syn-

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[2]
ESCON	160 Mb/s	STS-1-4v / VC-3-4v	193.536 Mb/s	1	5.11 Mb/s / 24.8 Mb/s	6.76 Mb/s
Fiber Channel	850 Mb/s	STS-3c-6v / VC-4-6v	898.56 Mb/s	13	412 kb/s / 85.82 Mb/s	2.415 Mb/s
Gbit Ethernet	1.0 Gb/s	STS-3c-7v / VC-4-7v	1.04832 Gb/s	95	281 kb/s / 1.138 Mb/s	376.5 kb/s

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 best-case client management payload bandwidth assumes 8 "payload" bytes per Client Management frame and the best-case residual overhead bandwidth conditions.

Table 1. *Virtually concatenated channel sizes for various transparent GFP clients.*

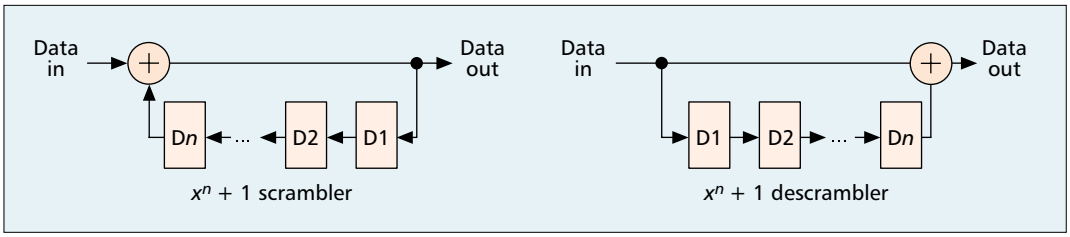

Figure 5. *Payload self-synchronous scrambler.*

chronized scrambler, which is reset at regular intervals based on the SONET/SDH/OTN frame, is adequate to defend against normally occurring user data patterns. It would be possible, however, for a malicious user to choose a packet payload that is the same as the frame-synchronized scrambler sequence. If this packet lines up in the correct position in the transport frame, an adequately long string of 0s or 1s can be generated to cause a loss of synchronization at the receiver. The resulting loss of synchronization will take down the transport link while the receiver attempts to recover, thus denying the link to other users in the meantime. This problem was originally discovered in ATM networks and is exacerbated by the longer frames used in POS or GFP. In order to guard against such attacks ATM, POS, and GFP use a self-synchronous payload scrambler to further randomize the payload data. This self-synchronous scrambler uses a polynomial of $x^{43} + 1$, which means that each bit of the ATM/POS/GFP payload area is exclusive ORed with the scrambler output bit that preceded it by 43 bit positions, as shown in Fig. 5. (The scrambler state is retained between successive GFP frames.) The decoder's descrambler reverses this process.

In order to use the same payload scrambling technique for both frame-mapped and transparent GFP, all of the GFP payload bits including the GFP-T superblock CRCs must be scrambled. As a result, the superblock CRC has to be calculated over the superblock payload bits prior to scrambling and checked at the decoder after descrambling. The drawback to a self-synchronous scrambler, however, is that each transmission error results in a pair of errors (43 bits apart here) in the descrambled data, which means that the superblock CRC must cope with this error multiplication. It has been shown [5, 6] that a CRC will preserve its error detection capability in this situation as long as the scrambler polynomial and the CRC generator polynomial have no common factors. Unfortunately, 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 was required that preserved the triple error detecting capability (which is the maximum achievable over this block size) without having any common factors with the scrambler. In order to perform single error correction, the syndromes for single errors and double errors spaced 43 bits apart must all be unique [6]. The code selected for the superblock is $x^{16} + x^{15} + x^{12} + x^{10} + x^4 + x^3 + x^2 + x + 1$, which has both these desired properties, and hence retains its triple error detection and optional single error correction capabilities in the presence of the scrambler [6–8].

Transparent GFP Client Management Frames

CMFs have the same structure as GFP client data frames but are denoted by the payload type code PTI = 100 in the GFP payload header. Like GFP client data frames CMFs have a core

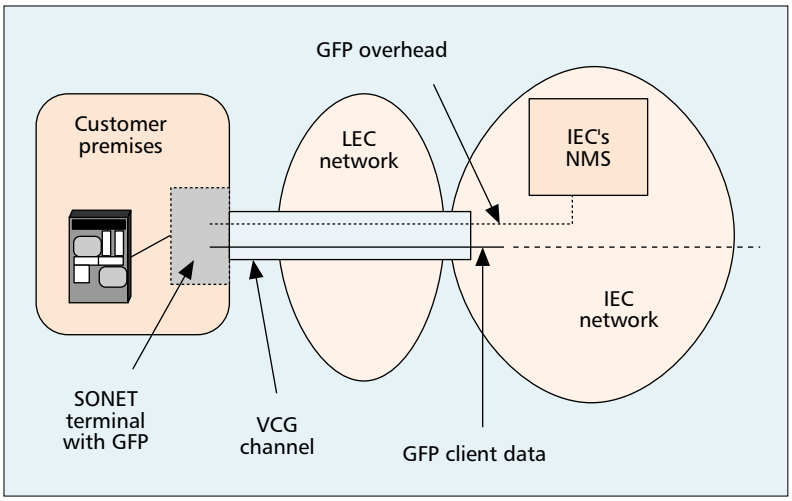

Figure 6. *An SDCC tunneling application example with transparent GFP.*

header, payload header (both with 2-byte header error checking, HEC) and an optional 32-bit FCS. The total CMF payload size in GFP-T is recommended to be no greater than 8 bytes.

Assuming an 8-byte payload area along with the 8 total bytes for the mandatory core and type headers, the payload efficiency will be 50 percent. Use of FCS and especially extension headers will greatly reduce the efficiency of the CMFs.

As noted above, there is some residual "spare" bandwidth in the SONET/SDH channel for each of the client signal mappings. As shown in Table 1, the amount of this "spare" bandwidth 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 this section and [9]. CMFs are also used for downstream indication of client signal fail.

Client Signal Fail Indication — GFP uses CMFs to indicate client signal fail (CSF) to the far-end GFP equipment. On detection of a failure defect in the ingress client signal a GFP CMF is transmitted following the current frame. This CMF uses PTI = 100, PFI = 0 (no FCS), appropriate EXI field, and UPI = 0000 0001 (loss of client signal) or 0000 0010 (loss of client character synchronization). Since the payload length indication is transmitted at the beginning of the client data frame and the onset of CSF can occur in the middle of a GFP client data frame, the remainder of the current client data frame will be filled with 10B_ERR codes to give it the required length.

CMFs indicating CSF are sent every $100 < T < 1000$ ms in order to prevent:

- The receiver from being overwhelmed with frequent CSF indications
- In the case of a frame multiplexed scenario, excessive hogging of the bandwidth for CSF indication of just one channel

Upon reception of the CSF indication, the GFP client sink adaptation process (GFP receiver) declares a sink client signal failure and outputs either 10B_ERROR or other illegal 8B/10B codes on the client egress signal. If the CSF condition is a loss of signal and lasts beyond some time limit, the client egress output signal transmitter (e.g., the laser) may be turned off. The CSF condition at the receiver is cleared by:

- Reception of a valid client data frame
- After failing to receive N CSF indications in $N \times 1000$ ms (a value of 3 is suggested for N)

Far-End Performance Reporting — The most universal client management application currently under consideration is the reporting of client-specific performance information from the far end of the GFP link. The GFP receiver can report such client-specific performance statistics as the bit error rate (BER) or ratios of good to bad client frames either on a periodic basis or upon being 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 can be valuable if one of the ends is in an unmanned office or 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, the potential exists for sending GFP-specific provisioning commands using CMFs. It is not uncommon for interexchange carriers (IECs) to provide the customer premises equipment (CPE) and rely on a local exchange carrier (LEC) to provide the connection between the CPE and the IEC network (Fig. 6). Ideally, the IEC would like to manage the CPE as part of its own network, which frees the customer from having to manage the equipment and allows the IEC a potential revenue source from providing the management service. Normally, management information is communicated through a SONET/SDH section data communication channel (SDCC). In order to prevent unwanted control access, however, carriers do not allow SDCC data to cross the network interfaces into their networks; hence, there is currently no way for the IEC to exchange management communications with the CPE through the intervening LEC network. GFP-T CMFs, however, provide a mechanism to tunnel the SDCC information through the intervening network. Table 1 shows the maximum amount of payload capacity that can be derived from the CMFs for SDCC tunneling or other operations, administration, and maintenance (OAM) applications, with the assumptions stated in the table notes and assuming a 20-byte CMF with an 8-byte payload field. For all client signal types, there is adequate bandwidth available to carry a 192 kb/s SDCC channel.

Potential Extensions to Transparent GFP

Three types of extensions are possible for GFP-T:

Other client signal types — The ETSI digital video broadcast standard was recently proposed as a GFP-T mapping, and Infiniband is another potential LAN signal. 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].

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, of course, require the definition of a GFP physical layer.

Other client management applications — Two applications have been described in the preceding section that make use of the versatile CMFs. Other extensions could include using the CMFs for a trace function to guarantee the correct connectivity if the GFP-T stream passes through a network element that routes GFP signals.

Conclusions

Transparent GFP provides an efficient mechanism for mapping constant bit rate block-coded data signals across a SONET/SDH network or OTN. Performing the mapping on a client character basis rather than a client frame basis significantly reduces the transport latency to a fixed number of bytes rather than a whole client frame, which is a critical issue for SAN protocols including Gigabit Ethernet. The translation of client block codes into more efficient 64B/65B mapping provides significant bandwidth efficiency increase while the superblock structure provides robustness. Transparent GFP also allows increased performance monitoring capability for the transport layer, and the ability to tunnel SDCC management information through an intervening network provides a powerful extension to network providers' capabilities.

References

[1] ITU-T Rec. G.7041/Y.1303, "Generic Framing Procedure," S. Gorshe, Ed.
[2] ITU-T Rec. G.707/Y.1322, "Network Node Interface for the Synchronous Digital Hierarchy (SDH)."
[3] ANSI T1.105, "Synchronous Optical Network (SONET) — Basic Description including Multiplex Structure, Rates and Formats."
[4] ETSI EN 300 417-9-1, "Synchronous Digital Hierarchy (SDH) Concatenated Path Layer Functions."
[5] T1X1.5/2001-094, "Impact of x43 + 1 Scrambler on the Error Detection Capabilities of Ethernet CRC," std. contrib. from N. Figueira, Nortel Networks, Mar. 2001.
[6] S. Gorshe, "CRC-16 Polynomials Optimized for Applications Using Self-Synchronous Scramblers," to appear, *Proc. ICC 2002*.
[7] T1X1.5/2001-125, "Recommended CRC-16 Polynomial for the Transparent GFP Superblock and Associated New Text," std. contrib. from S. Gorshe, PMC-Sierra, June 2001.
[8] T1X1.5/2001-174, "Optimum CRC-16 Polynomial for the Transparent GFP Superblock," standard contribution from S. Gorshe, PMC-Sierra, Sept. 2001.
[9] T1X1.5/2001-148, "Proposed Draft Text for GFP OAM Frames," std. contrib. from T. Wilson (Nortel Networks), M. Scholten (AMCC), and S. Gorshe (PMC-Sierra), June 2001.
[10] T1X1.5/2001-177, "Proposed ITU-T Contribution for Adding a 4B/5B Ethernet Mapping to Transparent GFP," std. contrib. by P. Thaler (Agilent) and S. Gorshe (PMC-Sierra), Sept. 2001.

Biographies

STEVEN S. GORSHE [SM'91] (steve_gorshe@pmc-sierra.com) received his B.S.E.E. degree from the University of Idaho in 1979 and his M.S.E.E. degree from Oregon State University in 1982, where he is also currently a Ph.D. candidate. He has been involved in applied research and the development of transmission and access systems since 1982. After beginning his career in GTE Lenkurt's Advanced Development Group in 1982, he moved to NEC America in 1988. From 1988 to 2000 he worked as a principal engineer for NEC America and chief architect for NEC Eluminant Technologies. His work at GTE and NEC included ASIC design and system architecture for subscriber access and SONET/SDH multiplex systems. He joined PMC-Sierra in 2000 as a principal engineer in the Product Research Group, working with ICs for SONET and optical transmission and access systems. He is a past chairman of the IEEE Communications Society's Transmission, Access, and Optical Systems Committee, and currently co-editor for *IEEE Communications Magazine*'s Broadband Access Series. He also serves as chief editor for the ANSI T1X1 Subcommittee responsible for SONET and optical network interface standards, and has been technical editor for multiple standards within the SONET T1.105 standard series as well as ITU-T Recommendation SG15 G.7041/Y.1303 (GFP). He has 24 patents issued or pending and several published papers.

TREVOR WILSON (wilsont@nortelnetworks.com) received his B.Sc. (electrical and electronics engineering) in 1970 from Queen's University, Belfast, Northern Ireland. He was involved in the computer, aerospace, bioengineering, and electrical power distribution industries before joining Nortel Networks in 1994. His work in Nortel Networks has focused mainly on transmission networks and systems design, and he is currently a systems design and standards advisor in the Optical Networks division, based in Belfast. He represents Nortel Networks on ETSI, ITU-T, and ANSI TIXI subcommittees, and has contributed widely during the development of virtual concatenation (including LCAS) and GFP standards.

The translation of client block codes into more efficient 64B/65B mapping provides significant bandwidth efficiency increase while the superblock structure provides robustness.