Analysis of the Train Communication Network Protocol Error Detection Capabilities

February 25, 2001

Philip Koopman
ECE Department & ICES
Carnegie Mellon University
Pittsburgh, PA, USA
koopman@cmu.edu

Tridib Chakravarty
ECE Department
Carnegie Mellon University
Pittsburgh, PA, USA
tridib@ece.cmu.edu

Abstract

The Train Communication Network (TCN) has been adopted as an international standard for use in critical transportation applications on trains. This paper discusses the results of a general review of the specification for error detection properties as an important factor of overall system safety. In general, TCN has excellent error detection properties and is much more thoroughly specified in this regard than other embedded network protocols. The only significant recommendation for improvement is prohibiting the use of variable- or multiple-length frames for any particular frame ID value to guard against corruptions that can cause undetected changes in message lengths (current implementations use only single lengths, but this is not specifically required by the standard). Additionally, it is important that designers pay close attention to receiver circuitry to minimize vulnerability to "bit slips" that could cause phase shifting and resultant burst errors in received Manchester-encoded bit streams.

1. Introduction

Error detection is a crucial part of any network communication protocol. Unfortunately, no error detection scheme can detect all possible errors, and every such scheme has an associated cost in communication bandwidth. Thus, every class of application requires a tradeoff between error detection capability and bandwidth cost. While in most applications a standard protocol can be used that assumes a standard level of tradeoff, when creating a new protocol it is important to perform and document this tradeoff.

This paper analyzes such a tradeoff made on a new network protocol for use in trains, called TCN (Train Communication Network). TCN is an embedded real-time data network proposed for use on trains [IEC99], and consists of two different networks with somewhat different protocols. The Multi-function Vehicle Bus (MVB) protocol is used for networks within a single vehicle (*e.g.*, a rail car), while the Wire Train Bus (WTB) is used across the length of an entire train. The TCN standard document has been prepared under the auspices of Working Group 22 of IEC Technical Committee 9: electric railway equipment. A complete description of MVB and WTB operation is well beyond the scope of this paper; readers are referred to the standard [IEC99] or to [Kirmann01] for operational details.

In safety critical transportation applications, the network must provide some defined minimum level of message frame transmission integrity, forming a solid foundation upon which other mechanisms may be added as needed for critical tasks. (We use the term "frame" for transmissions on the network because the term "message" has a specific meaning within TCN to be a data item that may be spread across multiple frames.) One component of this integrity is simply ensuring that a sufficient number of uncorrupted frames are delivered to perform required functions. But a second concern is ensuring that the probability of a corrupted frame being undetected is extremely low, presenting a quantified and acceptable level of risk to applications.

Detecting every possible corrupted frame is inherently impossible because any detection technique can succumb to a set of bit errors that, by chance, mimics an incorrect but seemingly error-free frame. In order to make the probability of such an occurrence sufficiently low, the error coding scheme used must be analyzed for vulnerabilities, and the physical layer of the network must ensure an appropriately low overall bit error rate via an appropriate of choice of medium and shielding. Thus it is important to characterize the maximum permissible BER for a protocol to achieve satisfactory error detection performance.

This paper analyzes the frame encoding and error detection capabilities of the two protocols that are part of TCN. Section 2 is a summary of operation for the MVB. Section 3 discusses vulnerability to unde-

tected errors based on check sequence encoding. Section 4 quantifies the added benefit of checking for Manchester encoding violations to overall error detection. Section 5 analyzes vulnerability to corrupted start and stop bit patterns. Section 6 discusses vulnerability to burst errors caused by receiver bit slippage. Section 7 discusses the WTB. Section 8 discusses other areas of protocol design that promote dependable operation. Finally, Section 9 presents conclusions.

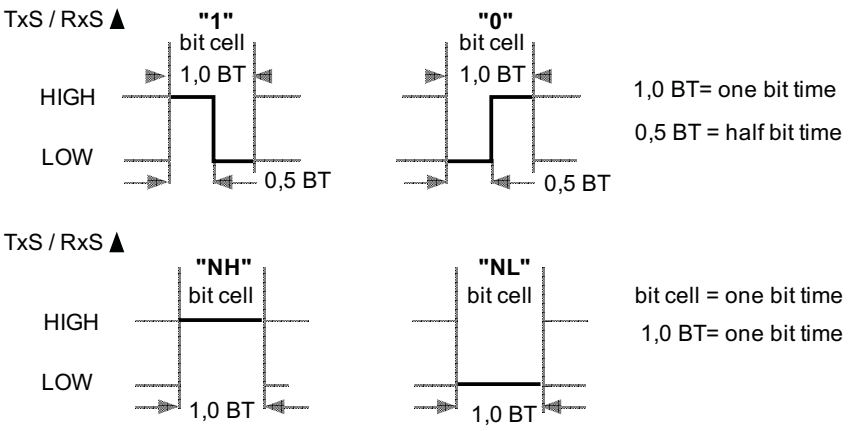

Figure 1. MVB bit encodings (from [IEC99]).

2. MVB characteristics

The purpose of this paper is to analyze the effectiveness of detecting transmission errors due to corruption from network transmission noise. The primary mechanisms for detecting such errors are observing bit encoding errors and detecting mismatches between the contents of a frame and the transmitted cyclic redundancy code (CRC) sent with the frame.

Manchester bit encoding (Figure 1) is used for the bits in the MVB frames, with four possible bit encodings: "1" has the first half-bit high and the second half-bit low; "0" has the first half-bit low and the second half-bit high; "NH" has both half-bits high, and "NL" has both half-bits low. Only the "0" and "1" symbols represent valid data. "NH" and "NL" are used to as marker bit values to uniquely encode frame start and end delimiters.

Figure 2 shows the general format of a frame on the MVB. A Start Delimiter preamble of 9 bits (including 1 start-of-transmission sync bit) provides a distinctive waveform by including NH and NL bits. Frame data is included in from one to four data payload sections, with each payload being 16, 32, or 64 bits in size. Frames with more than 64 bits of data are broken into multiple 64-bit data payloads as shown. Each data payload section is protected by an 8-bit Check Sequence (CS). The end of each frame is denoted by a 2-bit End Delimiter sequence comprising an NL bit followed by an NH bit. Frame length is inferred from the detection of an End Delimiter.

The MVB provides two primary types of error detection to detect errors caused by noise during transmission: invalid delimiter encoding and check sequence values. Both mechanisms can be augmented if receivers additionally detect frames with invalid Manchester bit encodings.

Start Delimiter (9 bits)	Data Payload (16, 32, or 64 bits)	Check Sequence (8 bits)	End Delimiter (2 bits)

Format for 16, 32, and 64-bit messages

Start Delimiter (9 bits)	Data Payload (64 bits)	CS (8 bits)	Data Payload (64 bits)	CS (8 bits)	End Delimiter (2 bits)

Format for 128-bit messages

Start Delimiter (9 bits)	Data Payload (64 bits)	CS (8 bits)	Data Payload (64 bits)	CS (8 bits)	Data Payload (64 bits)	CS (8 bits)	Data Payload (64 bits)	CS (8 bits)	End Delimiter (2 bits)

Format for 256-bit messages

Figure 2. MVB message formats.

3. MVB Check Sequence Error Detection

The MVB employs a Check Sequence protecting every data payload segment of 16, 32 or 64 bits. The Check Sequence consists of a 7-bit Cyclic Redundancy Code (CRC) from IEC standard 60870-5 as well as an even parity bit computed over the CRC value. The CRC polynomial used is:

$G(x) = x^7 + x^6 + x^5 + x^2 + 1.$

The CS enables detecting transmission errors via the following method. When a data payload is prepared for transmission, a CS value is computed and inserted into the frame's CS field. When a frame is received (assuming that the Start and End delimiters are uncorrupted), a new copy of the CS is computed based on the received data payload contents. This new CS is compared against the CS in the received frame. If the received and computed CS values match, the frame is considered correct; if not then a transmission error has been detected and the frame is discarded as having been corrupted. Note that this process takes into account the effect of the parity bit in the CS as well as CRC performance. (While an 8-bit CRC could be much more efficient than the 7-bit CRC plus parity bit, the approach used was selected for the MVB for legacy reasons.)

Figure 3 shows CS effectiveness for the MVB. These are the results of Monte Carlo simulations for undetected frame corruptions with varying numbers of randomly flipped bits within frame payload and CS fields (corrupted start/end delimiters were not considered in these measurements). The CS encoding used by the MVB successfully detects all possible 1-bit and 2-bit errors. By this, it is meant that all possible situations in which a single data payload or CS bit has been flipped from 0 to 1 or 1 to 0 will result in a CS mis-compare indication of a corrupted frame. Similarly, all possible corruptions of exactly two bits are detected.

Based on the data in Figure 3 and the fact that increasing numbers of bits flipped in a single frame are increasingly unlikely, the MVB is most susceptible to 3-bit random bit errors for 16-bit payloads, with an undetected error probability of 0.004 (equal to 0.4%). Other simulation results show that for the maximum payload size of 64 bits the CS is similarly most vulnerable to 3-bit random errors and has an undetected error probability of 0.0059 (equal to 0.59%). However, as seen in the next section, Manchester encoding considerations make 16-bit payloads the limiting case, especially when considering that the master/slave polling technique used in the MVB guarantees that at least half of all network traffic consists of the 16-bit payloads used in master frames.

Simulation results presented in this section were validated by comparing two independent simulation implementations and comparing results for the CRC portion of CS operation with an analytic model.

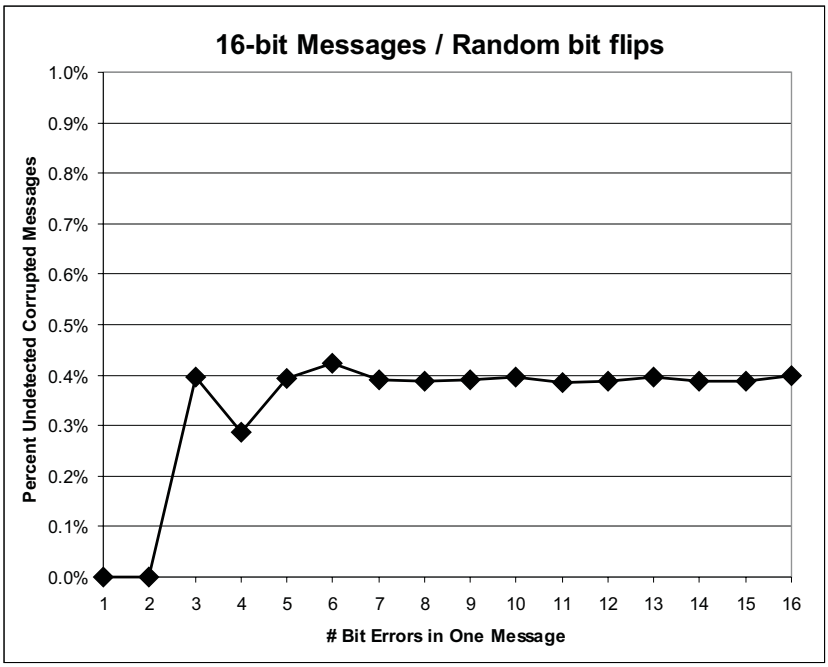

Figure 3. Undetected error rate for random bit flips with 16-bit payloads.

4. Using a Semi-Bit Encoding Error Model

The CS performance presented in the previous section uses a traditional "bit flip" fault model. While this model may be useful in NRZ (non-return-to-zero) bit encodings, use on Manchester encoded frames is highly questionable. This section of the paper discusses a more realistic error model based on semi-bits.

The problem with the bit flip error model is that Manchester encoded bits must be subjected to a sigmoid-shaped noise function to accomplish bit inversion. That is to say that accomplishing a bit value inversion requires that noise flip the first semi-bit in one direction and flip the second semi-bit in the opposite direction. This is a rather unlikely noise pulse to observe on an embedded network, rendering calculations based on Bit Error Rate (BER) very pessimistic for Manchester encoded networks.

Thus, we suggest the use of a semi-bit error model for computing more realistic undetected error rates. Instead of thinking of a high/low or low/high pair as a single Manchester-encoded bit, think of them instead as two independent but adjacent semi-bits. The error model then is that each semi-bit can be independently corrupted by being flipped, and that many such errors can be detected by checking for improper Manchester bit encoding prior to checking the CS value. A true bit flip occurs only when one valid Manchester encoded value is, by chance, converted to another valid Manchester encoded value via a pair of semi-bit inversions happening to hit the two halves of a single bit value. Only semi-bit errors paired in this way avoid detection by Manchester decoders and must then be detected with the CS value.

The semi-bit error model corresponds to thinking of the Manchester encoded network bit stream as actually being transmitted as pairs of NRZ semi-bits, with a guarantee that in a correctly encoded bit stream there are no more than two semi-bits of the same value adjacent to each other (excepting start and stop delimiters). Thus, we can now consider errors in terms of semi-bit flips of an NRZ network running at twice the speed of the original Manchester encoded network (two semi-bits per physical network bit).

For independent bit errors the performance of the CS field in the network being examined is dominated by the number of 3-bit errors that are undetected. The probability of that happening with a 16-bit frame is in turn dependent on the probability of having six semi-bit errors that just happen to result in three bit value inversions. Thus, the probability of undetected errors becomes a combination of the probability that all errors injected result in bit value inversion, and that the resulting inverted bits are undetectable by the CS.

For the case of three bit inversions in a 16-bit payload, there must be 6 semi-bit errors that occur in just the right pairing within the 35-bit length of a 16-bit payload frame. (We are assuming that semi-bit errors are independent, and they will have to by chance occur in just the right places to create 3 corrupted bits).

From Figure 2 the first 9 bits (which contain a total of 18 semi-bits) are the Start Delimiter. There are exactly two valid Start Delimiters, one for Master frames and one for Slave frames. These contain sequences of NL and NH values that are purposefully chosen to be quite different from each other, and differ in value by 13 semi-bits (giving a Hamming distance of 13 semi-bits). Getting just the right corruption pattern of 13 semi-bits is very improbable compared to the 6 semi-bit dominating case we are considering, and anything other than the two correct

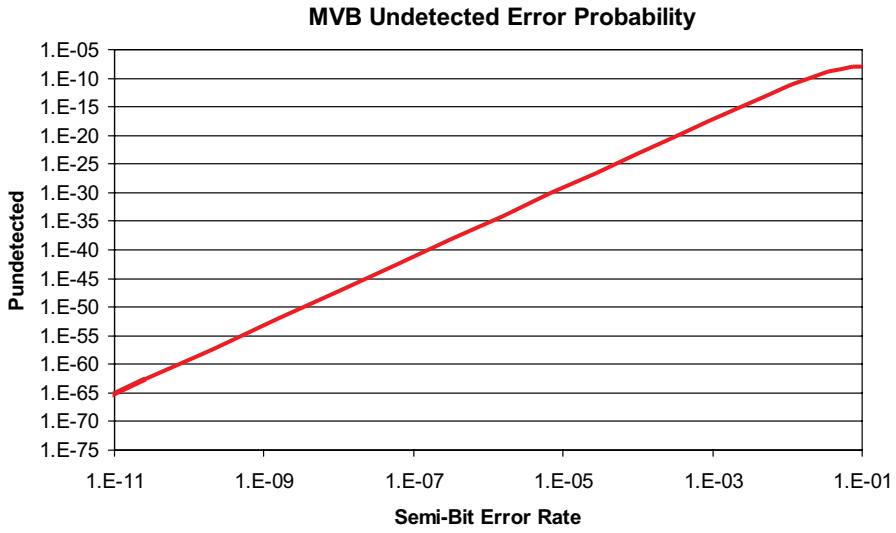

Figure 4. Burst errors/16 bit payloads semi-bit undetected error rate.

Start Delimiter values will result in the frame corruption being detected. Similarly, there is only one valid End Delimiter bit pattern.

Thus, to create an undetected error in this scenario it is required that *none* of the Start Delimiter nor End Delimiter bits be flipped. The chances of that occurring in a 35-bit frame are the chances that all of the six corrupted semi-bits will occur in the 24 bits of payload and CS fields:

$$\frac{24}{35}\times\frac{23}{34}\times\frac{22}{33}\times\frac{21}{32}\times\frac{20}{31}\times\frac{19}{30}=0.0829222$$

Given that all six semi-bit corruptions occur in the 24-bit payload/CS fields, there are 48 possible semi-bit positions that can be corrupted, giving the combinational value of $\binom{6}{48}$ possibilities. Of these, any semi-bit flip that is not paired with another semi-bit flip within the same full-bit boundary will be detected by the Manchester decoding logic, meaning that all six semi-bit flips must be paired into full bit boundaries. There are only $\binom{3}{24}$ combinations of 3 such full-bit flips possible. Thus, the probability of independent semi-bit flips happening to create clean Manchester-encoded bit flips with no Manchester encoding violations is:

$$\frac{\binom{3}{24}}{\binom{6}{48}}=\frac{\left[\frac{24!}{3!\times 21!}\right]}{\left[\frac{48!}{6!\times 42!}\right]}=\frac{42!\times 24!\times 6!}{48!\times 21!\times 3!}=0.000164935$$

Combining these two probabilities with the 0.40% undetected error rate of the CS itself gives an undetected failure probability of:

$$0.0829222\times 0.000164935\times 0.004=5.5\cdot 10^{-8}$$

This composite undetected error rate reflects the a priori probability that no semi-bit corruptions occur in the start delimiter, six semi-bit corruptions occur in the frame body in a way that actually flips three bit values without incurring a Manchester encoding violation, and that the resultant error is undetected by the CS (with probability 0.004). This value is significantly better than the CS protection of 0.004 alone. Longer frames would have lower undetected error rates because of the increasing likelihood of splitting pairs of semi-bit corruptions across a longer payload/CS span and thus are not the limiting case.

The other case of interest for a similar computation is a 64-bit payload with CS but no Start Delimiter, such as would be found in the second and subsequent payloads of a multi-payload frame. A similar computation involving the probability of 6 semi-bit data corruptions pairing to form a 3-bit corruption can be performed, and multiplied by a CS undetected error rate of 0.59% for 64-bit payloads. This yields a similar undetected error rate value of:

$$\frac{138!\times 72!\times 6!}{144!\times 69!\times 3!}\times 0.0059=3.2\times 10^{-8}$$

To understand the impact of these undetected error rates, consider the 16-bit payload frame as the limiting case. (This is a reasonable conservative approximation since fully half of the network frames are 16-bit Master frames and 16-bit frames have the highest undetected error rate for independent bit errors.) Any fewer than 6 semi-bit corruptions are always detected, as are any odd number of semi-bit corruptions.

Figure 4 shows the probability of undetected error given an overall semi-bit error rate. This figure accounts for corruptions of 6 and 8 semi-bits (more than 8 semi-bits is of small enough relative probability that such cases form a negligible effect).

If we take an extreme limit of 28,571 frames per second as the traffic load (back-to-back 35-bit frames with no gap time, which could not happen in a real system), there can be no more than 9.03 x 10^{11} such frames transmitted on an MVB in one full year of continuous operation. If we assume that we want no more than 10^{-6} undetected errors per year of operation (a typical aviation number for critical systems) then the probability of undetected error per frame must be 1.10 x 10^{-18}. From the computations for Figure 4 and this source of potential undetected errors, this means that the semi-BER on the network should be no worse than $7x10^{-4}$, which is quite a high bit error rate for a shielded network cable and is probably higher than will be encountered in reasonable field installations. Similar approaches can be used under other assumptions of undetected error rate requirements.

It is important to note that this analysis assumes the pessimistic case of a bit inversion medium. If the failure mode of interest is bit erasure (perhaps on a fiber optic network), undetected corruption of a Manchester-encoded bit is impossible, since erasure would force any pair of semi-bits to the same value. However, bit slip might still be possible in such media as discussed in Section 6.

5. MVB Delimiter-Based Error Detection

There are other possible sources of undetected errors beyond inverted payload and CS bit values. It is also possible that bit value errors will occur in the Start or End Delimiters, causing mis-interpretation of frame meaning or payload length.

From Figure 2 the first 9 bits (which contain a total of 18 semi-bits) are the Start Delimiter. As discussed in the previous section, getting just the right corruption pattern of 13 semi-bits to convert one Start Delimiter into another one is very improbable. Anything other than the two correct Start Delimiter values will result in the frame corruption being detected.

It is also possible for a data stream to be converted to a Start Delimiter by flipping 4 semi-bits in just the right positions (each valid Start Delimiter has two NH and two NL bits within a set of 8 otherwise valid bits after a one-bit Start Bit). However, for this to occur would require just the right data values to be corrupted in just the right way, and would further require the following bits to form a correct frame with matching CS value. This is very unlikely to happen, especially if frame lengths are checked as suggested below.

A potentially more serious problem would be an undetected corruption that results in a false End Delimiter that truncates a frame or a missing End Delimiter followed by system noise that resembles bit patterns enough to result in an overly long frame.

The End Delimiter consists of an NL bit followed by an NH bit, with end of frame being triggered at a receiver near the end of the NL bit (the subsequent NH bit provides, among other things, a balanced signal to avoid DC bias problems for coupling transformers). Because it would take only a single semi-bit flip to convert either a "1" or "0" bit to an NL bit, creation of an early End Delimiter via line noise would seem to be quite likely in operation. Additionally, the MVB specification requires ignoring any pulses after an End Delimiter, so the fact that an end of frame is followed by more data bits is sure to be ignored.

To compute the probability of a false End Delimiter going undetected we assume that a single semi-bit is flipped to create an NL bit. For a random semi-bit error in the frame payload and CS fields, this will happen with probability of 0.5. Given that there must be at least 8 valid data bits beyond the start field preceding the end delimiter to provide a well-formed CS field, this means that in a 16-bit payload only 16 of 35 bits can be corrupted to form a premature End Delimiter. That frame must still escape detection by the CS field, with probability of 0.004. Thus, given a single semi-bit error, the a priori probability of a premature End Delimiter is:

$$0.5 \times \frac{32}{70} \times 0.004 = 0.00091$$

This means that it is likely that premature End Delimiters will occur with even a low semi-bit error rate. 32-bit and 64-bit payloads would of course be even more vulnerable.

Because the MVB frame format is vulnerable to false End Delimiters caused by semi-bit errors, it is vital that systems check for correct frame lengths. At the protocol level this can be accomplished by generating an error condition if any frame does not have a payload length of exactly 16, 32, or 64 bits. The MVB standard requires that frames of incorrect length be ignored. Current implementations of MVB chips permit only a single frame size for each possible 12-bit frame ID value. However, a future version of the standard should specifically forbid accepting multiple different sizes for a 12-bit frame ID value to prevent compromising the current approach to length checking, since there is a non-trivial probability that a 64-bit or 32-bit frame will be truncated to 32 or 16 bits and pass all MVB error checks. This recommendation is largely a matter of formalizing current practice to make sure that future designers understand the importance of this design choice.

6. MVB Burst Error Detection

Because the MVB uses Manchester encoding, it might be vulnerable to burst errors. A burst error is defined as a contiguous stream of bits that have been wholly or partly corrupted. Burst errors can be caused by bursts of severe noise, and are very well detected by CRCs as long as the burst length is smaller than the CRC length.

However, with Manchester encoding, an additional source of burst error vulnerability is if a receiver "slips" by half a bit and interprets the incoming bits 180 degrees out of phase. That sort of bit slip might possibly be caused by a fairly brief noise disruption depending on the exact implementation of the receiver circuit. We do not know of a reasonable way to predict the probability of a slip occurring without extensive analysis of a particular receiver circuit. However, it is useful to understand the results of such a slip should it occur to motivate the requirement for receivers to avoid such slippages.

A bit slip during Start Delimiters will cause the frame to be ignored as invalid. However, a bit slip that occurs in the payload or CS fields might possibly result in a large number of received bit values being flipped over a length that exceeds the perfect error detection region of the CS, which is limited to 7-bit bursts due to the use of a 7-bit CRC field. Note that when a bit slip occurs, all subsequent data bits are inverted in value as a property of Manchester encoding.

To evaluate the effects of a bit slip we need to introduce the concept of P_{slip}, which is the probability that a bit slip will occur. P_{slip} is presumably related to the general bit error rate, but would need to be determined experimentally for a given system.

Given that a bit slip has occurred, there are two ways in which a frame error can go undetected. The first way is if a pair of slips compensate for each other and leave the total number of bits equal. The second way is if a bit slip deletes or inserts a bit value and is paired with a corruption of the End Delimiter that compensates for the bit slip in preserving received frame length.

If a pair of bit slips succeeds in injecting a burst error that compensates for length, the probability of detecting that burst error is approximately 0.004 for bursts of longer that 7 bits. For a 64-bit frame this represents a substantial vulnerability. For this reason it is important that if bit slips cannot be prevented, at least receiver designs should be biased so that they are very unlikely to permit both slips ahead and behind. If only slips ahead or only slips behind can occur, then frame length checks (recommended in the previous section) can eliminate this vulnerability.

If a bit slip is paired with a bit corruption that moves the apparent End Delimiter, then the a priori probability of undetected error is 0.004 if the bit slip happens more than 7 bits from the End Delimiter, which is likely for 64-bit frames and still reasonably possible for 16-bit frames. For this reason it is desirable to avoid the possibility of bit slips altogether by using careful tracking of bit times in receiver implementations. Because it is difficult to relate P_{slip} to the BER, it is difficult to produce a conclusive vulnerability estimate.

7. WTB Vulnerabilities

The WTB is based on the HDLC protocol (ISO 3309 and ISO 4335 standards), but is Manchester encoded instead of NRZ encoded. Because of the use of Manchester encoding, many of the vulnerabilities of HDLC are avoided. In general, the potential vulnerabilities of the WTB are similar to those of the MVB, except that they are less likely to occur.

Although the CCITT CRC polynomial used by HDLC is not necessarily optimal for short frames, it is a widely used standard and is more effective than the shorter MVB CRC polynomial. The use of a 16-bit CRC decreases the probability of undetected burst errors from 0.004 for MVB to approximately 0.000015 for the WTB for long bursts. The WTB can detect all burst data errors up to 16 bits in length due to the use of a 16-bit CRC.

The WTB specification requires a length match of frame size to the length field. This helps reduce the effects of bit slippage, although there is still a possibility of both a corrupted length field and a compensating end delimiter corruption. For this reason applications should check that the type of frame is consistent with the expected frame size, and ignore frames where this consistency check fails as well as ignore frames of unknown type for which there is no a priori length information available. Additionally, as with the MVB, each frame type should be restricted to permit only a single valid frame length.

8. General Design Points

In addition to the design points discussed in preceding sections, it is worth noting that the TCN specification has done a thorough job in dealing with a wide variety of design issues. Most of these issues are typically ignored or glossed over in other comparable protocol specifications for other domains, so their presence is an indication of the high level of attention paid to robustness and dependability in the TCN protocol. The most noteworthy features include:

- An MVB freshness counter on periodically refreshed variables to detect when variables may have gone stale
- A very thorough procedure for reconfiguration including specific consideration of timing ties/races
- Specific provision for media redundancy (this is done with other protocols, but is often an after-the-fact addition that is not part of the protocol standard).
- Provision of fritting on the WTB to ensure good connections.

In addition to the above strengths, there is an area of possible concern that has become an issue in other domains: correlated failures on redundant media. Many TCN-based systems will be constructed with redundant physical media and dual receivers. In some physical installations media may be close enough together that common mode disruptions will cause identical or correlated failures in received data. Designers should be cautioned that great care must be taken in assessing a design before assuming that noise-induced failures will result in different data on redundant lines. (There is very little information available on this topic, but designers are well advised to realize that it is a possible issue.)

9. Conclusions

We have studied the effectiveness of error detection codes via simulation and analysis for the TCN network protocol. A particular innovation in this paper is the use of a semi-bit error model for the analysis of vulnerability to undetected errors for Manchester encoded data streams.

Based on our experience, the TCN design is significantly more robust than typical embedded networks such as CAN or LonTalk. Specific attention has been paid to a wide variety of problems that have histori-

cally been problems in other networks. However, there are a few areas that could be further improved via tightening the specification or including appropriate cautions. The standard should prohibit permitting multiple different frame lengths for any particular frame ID value to reduce the vulnerability to noise-induced false End Delimiter errors and any potential bit slip burst errors for the MVB and the WTB. While it would be obvious for an implementer to detect Manchester encoding phase violations and report them as frame corruptions, there is apparently no explicit requirement to do so; one should be added to the specification for both MVB and WTB. Additionally, it would be helpful to include cautions about possible correlated local errors on redundant media.

Beyond this work, the issue of characterizing the probability and results of bit slip errors remains open. It is important that receivers in TCN implementations be highly immune to bit slips to avoid undetected burst errors in frames.

Finally, it should be realized that every network protocol has its limits. While the use of Manchester encoding significantly improves the capabilities of the relatively short CRC fields used in the MVB and WTB, there is still a small chance that data errors will go undetected. It is recommended that future work include deploying an overlay protocol for frames that includes very stringent error detection mechanisms, including at least a 32-bit CRC for use in safety critical frames (*e.g.*, [Krut96]). However, that can be done without modifying the current TCN standard, and is a separate issue. And, of course, it should be realized that at some point the network will have such high integrity that the dominant vulnerabilities for undetected errors will be found in components beyond the network, such as the datapaths of microcontrollers or the network interface circuitry.

10. Acknowledgments

Thanks to Hubert Kirrmann of ABB Corporate Research and Pierre Zuber of ADtranz for their support and helpful advice. This work was funded by ADtranz Corporation and PITA. PITA is the Pennsylvania Infrastructure Technology Alliance, a grant sponsored by the Pennsylvania Department of Community and Economic Development. Equipment support was provided by Intel, Compaq and Microsoft.

11. References

[IEC99] IEC 61375 Electric Railway Equipment - Train Bus - Part 1: Train Communication Network, 1999.

[Kirrmann01] Kirrmann, H. & Zuber, P., “The IEC and IEEE Train Communication Network”, *IEEE Micro*, March/April 2001 (in press).

[Krut96] Krut, Gary, *Justification for the Format of Safety Telegram*, ADtranz corporation technical document, 1996.