Backward-compatible Enhancements to the 10 Mb/s Single-Pair Ethernet on Shared Media

by

HUSZÁK Gergely György

A dissertation submitted for the degree of
Doctor of Philosophy in Engineering (Ph.D. Eng.)

University of Electro-Communications (UEC)
Tokyo, Japan

15th March 2022

共有メディア方式における
10 Mb/s シングルペアイーサネットの
後方互換性をもつ機能拡張に関する研究

フサーク　ゲルゲイ　ジョルジ

博士論文（工学）

電気通信大学（東京）

令和4年3月15日

Dissertation Committee Members at the University of Electro-Communications

Professor YOSHINAGA Tsutomu

Professor MATSUURA Motoharu

Associate Professor OGAWA Tomohiro

Associate Professor WU Celimuge

Professor Emeritus MORITA Hiroyoshi

Associate Professor OHZAHATA Satoshi

Contents

Chapter 1 Introduction

1.1 Background

Ethernet is a family of standard wireline communication networking [1] technologies over both conductive and optical media. It has been, and still is, among the most innovative and future proof **wireline technologies** in active development. One of the key attributes of this development has been not losing sight of the importance of maintaining interoperability in a very large family of wired and wireless networking technologies, that successfully managed to bridge over 30 years of advancement.
The focus on interoperability is a key concept, as it allows inter-connecting a very diverse set of various systems through the following notable means:

- As for the **inter-system** interface, the following features of Ethernet are the common denominators:
  - The **Ethernet frame** (1) allows a single Local Area Network (LAN) to seamlessly interconnect (bridge) various devices, such that the frame emitted by the original sender is verbatim to the frame(s) delivered to the receiver(s), such as:
    - Personal computers of all kinds;
    - Servers, printers, and other peripherals;
    - Internet of Things (IoT) devices implementing Machine-to-Machine (M2M) communication with other control or monitoring functions of higher complexity, located arbitrarily far from the points of acquisition or actuation.
  - The **Ethernet Media Access Control (MAC)** enables the inter-connection of stations over links of various properties (with respect to speed, duplexity, etc.) in a way that provides reliability and probabilistic fairness even over Half Duplex (HD) connections, through the utilization of features like Carrier Sense Multiple Access with Collision Detection (CSMA/CD).

    Additionally, the MAC also provides versatile uni-, multi-, and broadcast addressing methods over a structured 48-bit address space, that enables even devices with lower capabilities to be connected to arbitrarily high bandwidth segments;
  - **Auto-Negotiation (AN)** [3] allows physical devices with different features and capabilities to automatically, quickly and reliably establish a link by mutually agreeing on a common set of attributes and features essentially serving as a “common language” for the link;
  - Last, but not least, the **Frame Check Sequence (FCS)** used by the Ethernet frame provides a high level of relative error detection capability, that follows the *miss at most once in a lifetime of the universe* design principle.
- Regarding the **intra-system** interfacing, Ethernet utilizes the Media Independent Interface (MII) that enables a station to be connected to any of a large set of Physical Layer Devices (PHYs) without need for a change to the higher

(1) IEEE Std 802.3-2018 [2] “3. Media Access Control (MAC) frame and packet specifications”.

layers. For example, a station being upgraded from a lower- to a high-speed, or moved to a different type of Ethernet network segment, needs only a new PHY, then it is expected to face no interoperability-related difficulties along the process due to the standard interface provided by MII.

While Ethernet has originally been a LAN technology, it clearly covers part of the Metropolitan Area Network (MAN) and Wide Area Network (WAN) communication needs, for example by one of the following means:

1. **Directly**: Many of the Ethernet PHYs provide a reach way beyond what is typically used by LANs. For example, 25GBASE-ER provides a reach of up to 40 kilometers;

2. Through **tunneling and encapsulation**, and other variants of these, referred to as Virtual Private Networking (VPN), Ethernet frame – while potentially crossing multiple networks – may travel arbitrarily far on the globe, relieving LANs from geographical and reach limitations;

3. Some specialized Ethernet solutions, for example the Ethernet Passive Optical Network (EPON) and its extensions, establish point-to-multipoint passive optical networking, often used for broadcasting.

Further, as the most common wireless technologies, such as Wi-Fi and Bluetooth, natively support Ethernet frame encapsulation, the effective reach of Ethernet also goes beyond the domain of wireline communication.

Until recent times Ethernet has been very well established in Information Technology (IT) networks and in some special segments of communications (for example automotive networks tailored specifically around car geometries and their bandwidth demands), but it has achieved limited penetration with Operational Technology (OT) use cases. In other words, until recently, no Ethernet PHY existed that could cover general-purpose industrial needs.
One of the cardinal reasons for this is the fact that industrial use cases have drastically different set of requirements than those of IT, including, but not limited to the following aspects:

- **Bound on the** channel access **delay** over HD links: Ethernet has not been able to provide this, due to CSMA/CD being the channel access method of Ethernet. In industrial communication, let alone control loops and functional safety-related use cases, it is essential to be able to calculate worst case one-way propagation and/or round-trip delays. While developments outside of the Institute of Electrical and Electronics Engineers (IEEE) Working Group (WG) 802.3 provided means to achieve better response times – generally referred to as Time-Sensitive Networking (TSN) – these typically assumed bridged Point-to-Point (Pt2Pt) links, and are still not ready for operate on HD Multidrop (MDROP) links;

- Gap between needed and available **bandwidth**: As later shown in Section 2.2, Ethernet clearly kept pushing ahead with extending offered bandwidth and while in case of IT networks this is being considered as a *blessing*, for systems with limited resources and much lower need for throughput this may become a *curse*. This is because a continuous influx of large amount of data can saturate the low-end stations, disrupting their normal operation that is required to provide critical functionality for the overall system;

- **Power budget**: In case of for example an office network, the power consumption of the networking components relative to that of the complete system is typically negligible or tolerable. For this reason high-speed PHYs often use modulations and protocols that consume relatively much power. For industrial, automotive or automation systems, power consumption is one of the most important metrics used during system design and components selection, for example:
  - For a battery-operated hazard detector located far away from the industrial plant, it is essential to remain operational without the need for frequent maintenance (battery replacement, etc.);
  - An industrial sensor operating based on energy harvesting must make utmost effort to limit its own power usage;
  - A car left at an airport garage shall maintain as much charge in its battery as possible, not to become a nuisance for its owner after a couple of days.
- And, as always, **cost and complexity** are among the main driving factors for systems that are produced in the millions and beyond.

Due to these, industrial, automotive, and automation OT networks have had their own ecosystem of networking technologies to build on, such as Controller Area Network (CAN), Local Operating Network (LON) specified by standard CEA–709.1–B, FlexRay, etc., over appropriate line-driver ("transceiver"), such as RS-485. These systems were inherently designed to follow the industry's approach mentioned above.

Finally, recent years have seen rapid development in these directions within the scope of Ethernet as well: a pair of new Ethernet PHYs, called 10BASE-T1S and 10BASE-T1L, have been standardized, forming the initial set of the 10 Mb/s Single-Pair Ethernet (10SPE) PHYs, which are expected to soon be followed by enhanced devices of similar (10BASE-T1M) and higher speeds (100BASE-T1L, 1000BASE-T1L). These technologies and devices have been, and still are being, developed with extended industrial, automotive, and automation use cases in mind, including covering requirements for power efficiency, low complexity, resilience and reliability, intrinsic safety, etc.

The research presented in the dissertation aims at helping the technology move forward even further in these directions, benefiting – among others – the industry.

1.2 Objectives and contributions of the dissertation

In the dissertation we are concerned with the research that has been conducted with the intention of improving the quality of the recently reborn **MDROP Ethernet technology**, that is now aimed primarily at industrial, automotive and automation use cases, without precluding use beyond these areas.
The objective of dissertation was to improve the noise resilience of the new Single-Pair Multidrop (SPMD) Ethernet PHYs, through the following two means:

1. Firstly, the target was to understand how good various Ethernet frame preamble candidates considered for one of the newest Ethernet PHYs – referred to as 10BASE-T1S – really were. Based of the outcome of this efforts, the aim was to propose better preambles that would allow implementation of new features, such as Operations, Administration, and Maintenance (OAM) and Out of Band (OoB) signaling;

2. Secondly, the aim was to implement a Forward Error Correction (FEC) method for the longer-reach successor of 10BASE-T1S – referred to as 10BASE-T1M – that provably meets a set of interoperability and backward-compatibility requirements. Achieving these would make 10BASE-T1M more suitable for industrial and automotive use cases that are characterized by the presence of high-energy impulse noise sources.

In the dissertation, these objectives are achieved by the systematic use of research and development methods, through the following steps:

- First, a detailed and unambiguous definition of the channel model used for Ethernet frame preamble research and evaluation is given;
- Through the application of this model, it is shown that earlier results with regards to the relative order of the Differential Manchester Encoding (DME)-based preamble `JJHH` and the Golay Complementary Sequences (GCSs) Ga32 and Gb32 were incorrect;
- While acknowledging that through the goodness evaluation method consensually accepted in IEEE Project 802.3cg [4] (P802.3cg), Ga32 and Gb32 are better than their DME-based counterparts, research shows that – contrary to the earlier conclusions – DME preambles better than `JJHH` do exist. The application of these better preambles also clears the way towards improving noise immunity of any future extensions of 10BASE-T1S, possibly in a backward-compatible manner;
- To improve the burst noise immunity of not only the Beginning of Frame (BoF) detection, but also that of the payload and the frame's End Sequence Delimiter (ESD), a novel backward-compatible Forward Error Correction (FEC) method – subsequently referred to as a $\{c, u\}$ coding scheme – is proposed.

  The backward-compatibility of the method is based on avoiding the appearance of certain 5B symbols – so-called Forbidden Symbols (FSs) – in the codeword. The proposed Forward Error Correction (FEC) consists of a unique combination of the following new techniques:
  - A linked-list-based method to transcode Forbidden Symbols (FSs);
  - A code construction method – referred to as FS Transcoding Recipe (FTR) – that extends the potential of the FS transcoding for cases where the linked-list would not be able to bridge two adjacent FSs;
  - An algebrbaic coding technique over finite fields, that allows FS avoidance in the parity symbols of the underlying Error Correcting Code (ECC).
- The dissertation also shows the bounds applicable to the constructability of an arbitrary $\{c, u\}$ coding scheme;
- During this analysis process, it is shown that, while multiple $\{c, u\}$ solutions do exist, the $\{19, 19\}$ coding scheme is the optimal one;
- Next, the dissertation presents how the performance of the proposed FEC may be extended through the application of well-known results, such as erasure correction and interleaving, making the $\{c, u\}$ coding scheme capable of providing **per-frame configurable impulse noise immunity** that is guaranteed to be **backward-compatible** with 10BASE-T1S;

- To demonstrate how an actual $\{19, 19\}$ coding scheme can be established, a step-by-step construction process is presented, starting from the definition of the base parameters (constants), through the method behind correctly partitioning 5B symbols, up to the application of the example scheme on an end-to-end encoding and decoding process of an Ethernet frame in the presence of error;
- After presenting a full example, the proposed solution is compared to other alternatives;
- Finally, the performance of the proposed coding scheme is evaluated through the analysis of its efficiency and incurred coding delays.

While majority of the work presented here is self-contained, and can be applied directly to 10BASE-T1S or other communication networks, in Section 5.2 the dissertation also points out areas of future interest to possibly further develop the proposed ideas and methods.

1.3 Organization of the dissertation

The dissertation is structured as follows:

- **Chapter 2** introduces the underlying technology based on which the research has been done. The introduction covers the history of Ethernet, relevant organizational aspects of IEEE, procedural features of standard development in IEEE WG 802.3, and technical considerations of the PHYs and other – closely related – devices specified by 802.3 and other organizations working in coordination with it.

  This chapter also discusses the development and standardization process that has led to the creation of the PHY called 10BASE-T1S. Sections throughout this chapter highlight the key features 10BASE-T1S combines in a way that enables it to be a core component of a large variety of wireline networks;
- **Chapter 3** discusses the first (earlier) theme of the research conducted on 10BASE-T1S, which led to the recognition of a small new set of Ethernet frame preambles that are superior to the one currently used by the standard. The aim of these is to increase the stationary noise resilience of the PHY;
- **Chapter 4** focuses on the second (latter) theme of the research, that realized a per-frame configurable impulse noise-resilient FEC scheme that meets a set of requirements that make it suitable for being used by the longer-reach successor of 10BASE-T1S, called 10BASE-T1M;
- **Chapter 5** summarizes the results presented in the dissertation and discusses possible future paths for the research;
- After the main chapters, the dissertation includes the **List of Symbols and Notations**, the **List of Terms and Abbreviations**, the **List of Equations**, the **List of Tables**, the **List of Figures**, and the **List of References**;
- Finally, **Appendix** presents all details of simulations used during the research, to allow verification and reproduction of all the results presented in the dissertation.

All references – in including those to the footnotes – operate as hyperlinks.

Chapter 2 The evolution of technology and research

2.1 History of Ethernet

Ethernet has been standardized by the Institute of Electrical and Electronics Engineers (IEEE) Working Group (WG) 802.3. The name of the WG originates from its start date (February 1980) and the numerical index "3" of the WG, which was formed for the sole purpose of maintaining and further developing the "IEEE Standard for Ethernet". At time of writing the dissertation the latest version of this standard is IEEE Std 802.3-2018 [2].
802.3 is an active WG that provides regular updates to the main body of standard text via the following two means:

- *Amendments*: New text manifesting as new clauses, as well as changes to existing clauses;

- *Corrigenda*: Correction of errors in the existing standard text.

Architecturally Ethernet fits into Layer 1 (Physical Layer) and Layer 2 (Data Link Layer (DLL)) of International Organization for Standardization (ISO)/Open Systems Interconnection (OSI) model and provides technologies for Local Area Network (LAN), Metropolitan Area Network (MAN), as well as for Wide Area Network (WAN) types of communication.

Ethernet support both Half Duplex (HD) and Full Duplex (FD) modes of communication over Point-to-Point (Pt2Pt) and Multidrop (MDROP) links, with the exception that FD mode over MDROP is yet nonexistent.

One of the definitive traits of Ethernet is that its Media Access Control (MAC) defined by Clause 4 (2) uses Carrier Sense Multiple Access with Collision Detection (CSMA/CD) media access method, during which physical collisions between transmitters are detected locally, and these cause each of the colliding transmitters to back-off and later retry their respective transmissions using the "truncated binary exponential backoff algorithm" (3).

In Ethernet, the interface between Layer 1 and Layer 2 is the Media Access Control/Physical Signaling (MAC/PLS) (4) and it supports various speeds ranging between 10 Megabit(s) per Second (Mb/s) and 400 Gigabit(s) per Second (Gb/s), while recent efforts aim at achieving the nominal rate of 1.6 Terabit(s) per Second (Tb/s). Further details of this architecture are depicted in the first figure in the standard (5), while in the dissertation Section 2.3 will show how this general architecture reflects in the new 10 Mb/s Single-Pair Ethernet (10SPE) Physical Layer Device (PHY) 10BASE-T1S.

2.2 The IEEE Project 802.3cg

Since its birth, Ethernet has had a main development trend that was characterized by a monotonic increase of throughput, to meet increasing demands posed by the large-scale spread of digital networked systems at different scales. During the most recent years, in parallel with this main trend, an additional set of use cases has been brought forward by various automotive, industrial, building automation, process

(2) IEEE Std 802.3-2018 [2] "4. Media Access Control".
(3) IEEE Std 802.3-2018 [2] "4.2.3.2.5 Collision backoff and retransmission (half duplex mode only)".
(4) IEEE Std 802.3-2018 [2] "Clause 6. Physical Signaling (PLS) service specifications".
(5) IEEE Std 802.3-2018 [2] "Figure 1–1—IEEE 802.3 standard relationship to the ISO/IEC Open Systems Interconnection (OSI) reference model" in "1.1.3 Architectural perspectives".

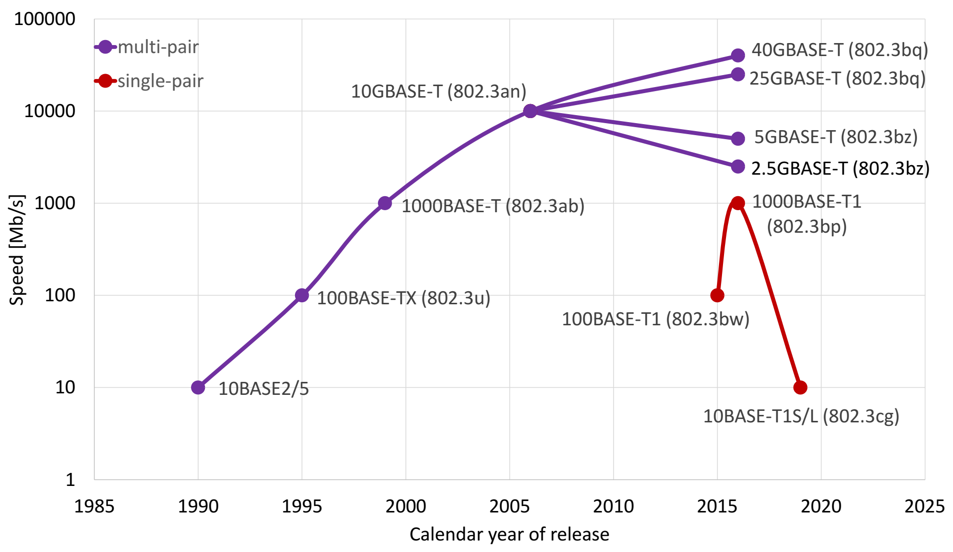


Figure 2.1: Status of the evolution of Ethernet speeds at the time of the approval of IEEE project P802.3cg

control, and in-system networking stakeholder. These have a drastically different focus, requirements and preferences, and favor lower costs, lower complexity, higher level of interoperability, and increased reliability over higher speeds that are otherwise superfluous in the given system.

These needs gave birth to the IEEE Project 802.3cg [4] (P802.3cg), initiated and run by a large selection of market leading entities, including, but not limited to, Cisco, Commscope, Aquantia, Pepperl+Fuchs, OmniPHY, Broadcom, Rockwell Automation, Phillips, Daimler, BMW, Phoenix Contact, and Endress+Hauser. The effort was soon joined by various universities from different continents, qualification houses, measurement technology and equipment manufacturers, and smaller, bleeding edge analog silicon technology developers that have been active in other areas of networking, such as Universal Serial Bus (USB), as well,. The evolution and relationship between various development branches of Ethernet at the time of the approval of P802.3cg is shown in Figure 2.1.

Throughout approximately three years of development effort made in accordance with the IEEE rules and processes [5] (that exist also to avoid domination and other distortions to the consensus), and following the pre-agreed and -approved objectives [6], P802.3cg yielded a series of drafts [7] as well as an introductory guide [8] to the new technology for the external viewer. The last revision of the drafts (d3.4) has been recommended for approval by IEEE Standards Review Committee (RevCom), and has subsequently been approved for publication by IEEE SA SB (SASB) in November 2019, resulting in the publication of IEEE Std 802.3cg-2019 [9]. After the conclusion of the project, introductory materials were also produced outside of IEEE [10].
P802.3cg established two new PHYs called 10BASE-T1S and 10BASE-T1L. The naming of these PHYs follows the naming convention of IEEE:

- `10`: 10 Mb/s;
- `BASE`: Baseband operation;

- `T`: Twisted Pair (TP) medium [(6)];
- `1`: Single-pair medium;
- `S`: Short(er)-reach;
- `L`: Long(er)-reach.

In one hand, as also apparent from the naming, these two PHYs share the following basic attributes:

- *Speed*: Both PHYs offer a nominal rate of 10 Mb/s at MAC/PLS;
- *Physical medium*: Both PHYs operate over a single balanced pair of conductive medium, that is typically, but not necessarily, TP copper wire.

On the other hand 10BASE-T1S and 10BASE-T1L have multiple essential differences as well:

- 10BASE-T1S is a **short-reach** (at least 15 or 25 m, depending on the mode [12]), ultra-low-complexity PHY, with an optional MDROP mode of operation and a Physical Layer Collision Avoidance (PLCA) technique that is capable offering bounded channel-access time and a total effective throughput of near 10 Mb/s even in saturation. The main body of specification of 10BASE-T1S is covered by Clause 147 [(7)], while Clause 148 [(8)] defines PLCA as a Reconciliation Sublayer (RS);
- 10BASE-T1L is a **long-reach** (of up to 1000 m), strictly FD over Pt2Pt PHY, supporting low-power operation. 10BASE-T1L is specified by Clause 146 [(9)].

2.3 10BASE-T1S

As explained in Section 2.2, 10BASE-T1S is a newly defined 10 Mb/s Baseband Single-Pair Ethernet PHY for short reach that supports MDROP. This section focuses on the new and reinvented features of this PHY.

While Section 2.1 introduced the general architecture of an Ethernet PHY in its entirety, how this manifests itself in case of 10BASE-T1S is depicted by the first figure in Clause 147 [(10)], the summary of which is shown in Figure 2.2.

Subsequent sections discuss the most relevant features of 10BASE-T1S from the perspective of this research.

[(6)] To incorporate the backplane requirements, during the standard development process, the root Project Authorization Request (PAR) [11] of P802.3cg has been amended by the final PAR [12], as part of which the specification has been relaxed and the requirement for "twisting" has been dropped (see Section 2.3.4 for more details on the relevance of this change).

[(7)] IEEE Std 802.3cg-2019 [9] "147. Physical Coding Sublayer (PCS), Physical Medium Attachment (PMA) sublayer and baseband medium, type 10BASE-T1S".

[(8)] IEEE Std 802.3cg-2019 [9] "148. PLCA Reconciliation Sublayer (RS)".

[(9)] IEEE Std 802.3cg-2019 [9] "146. Physical Coding Sublayer (PCS), Physical Medium Attachment (PMA) sublayer and baseband medium, type 10BASE-T1L".

[(10)] IEEE Std 802.3cg-2019 [9] "Figure 147–1—Relationship of 10BASE-T1S to the ISO/IEC OSI reference model and the IEEE 802.3 Ethernet model".

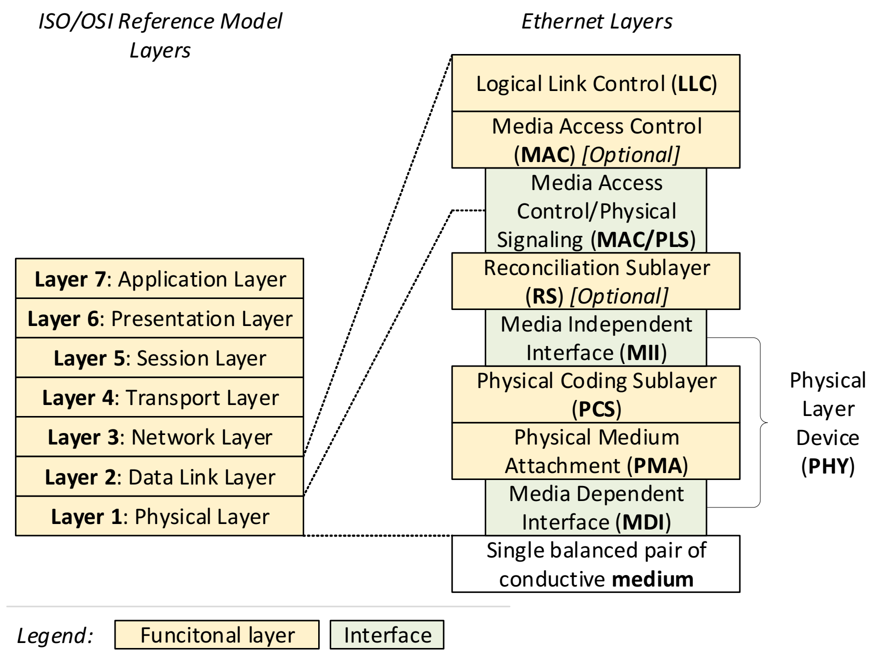

ISO/OSI Reference Model Layers
Ethernet Layers
Logical Link Control (LLC)
Media Access Control (MAC) [Optional]
Media Access Control/Physical Signaling (MAC/PLS)
Reconciliation Sublayer (RS) [Optional]
Media Independent Interface (MII)
Physical Coding Sublayer (PCS)
Physical Medium Attachment (PMA)
Media Dependent Interface (MDI)
Single balanced pair of conductive medium
Physical Layer Device (PHY)
Layer 7: Application Layer
Layer 6: Presentation Layer
Layer 5: Session Layer
Layer 4: Transport Layer
Layer 3: Network Layer
Layer 2: Data Link Layer
Layer 1: Physical Layer
Legend:
Funcitonal layer
Interface

Figure 2.2: Relationship between ISO/OSI layers and the Ethernet layers of a 10BASE-T1S-based system

2.3.1 Shared media: Multidrop

Some of the very first PHYs defined by IEEE Std 802.3-2018 [2] in the 1980s were 10BASE5 and 10BASE2. These are a.k.a. the "thick" and "thin" coaxial Ethernet", or "Thicknet" and "Thinnet" (both respectively). These 2 PHYs offered a 10 Mb/s nominal throughput over two different types of coaxial media, and both supported MDROP, allowing more than 2 stations to be connected to a single continuous conductive segment, referred to as "mixing segment" (as opposed to the segments that support Pt2Pt links, and that are known as "link segments").
MDROP buses have several notable advantages and disadvantages, as follows:

- Some of the main **advantages** are as follows:
  - *Complexity and costs*: The mixing segment is capable of interconnecting two or more stations without any active components beyond the Media Dependent Interface (MDI). Section 2.3.4 gives multiple use case examples, where this property of 10BASE-T1S alone makes it a preferred choice;
  - *Flexibility*: Networks built using Pt2Pt links and relay components (11) typically need careful advance planning (network design) and tend to give a more rigid response to changes. As an example, addition of new stations may require changes to, or even replacement of, the relay components, while a mixing segment may only need addition of copper wires;

(11) Hubs, bridges (switches), routers, etc.

   - *Resource need*: In addition to the physical resources mentioned right above, a segment that builds on MDROP may need less non-materialistic resources, such as energy (supply power);
   - *More efficient powering*: When power is supplied via the single-pair dataline of a MDROP network, the voltage drop of the supply is not more than what the conductive medium inherently inflicts due to its resistive and capacitive nature, which can also be adjusted through the proper selection of the cable properties, such as its structure, its thickness, or the material used for insulation.

     This however is not the case with a network built of a "chain" of Pt2Pt links, for example using Two-port MAC Relays (TPMRs) or more complex bridging elements that use Power over Ethernet (PoE). In such a case, each TPMR would need to be a Powered Device (PD) and a Power Sourcing Equipment (PSE) at the same time, causing considerable, non-recoverable loss to the limited power budget of the PoE system.

- Some of the main **disadvantages** are as follows:
   - *Global effect of component failures*: In a network consisting of Pt2Pt links, a failure effecting a single link – such as cable problems, termination issues, station failure [(12)], etc. – typically renders only the affected segment/station nonoperational (and "dangling" stations may remain accessible through redundant links), while in a mixing segment such failure may easily affect the performance of the whole segment, including the reachability of all its stations.

     It is worth noting however, that oftentimes single component failures may have similarly severe impact on a bridged network, if the failure affects the a bridge or a trunk [13] without redundant path;
   - *Lower total throughput*: In case the communication taking place in a network is partitioned such a way that subset(s) of stations are communicating primarily among each other, a switched network may operate so that the overall throughput of the network exceeds the capacity of its fastest link. Such apparent increase of throughput may not be achieved over a mixing segment, as there a single transmission prevents all other stations from transmitting. In addition, a network consisting of Pt2Pt links may utilize FD links with Time-Sensitive Networking (TSN), which is not currently possible in a mixing segment.

It is worth noting that some of the disadvantages listed here have well-known mitigation techniques, while those that do not are already in focus of ongoing research within, and also outside of IEEE.

As since 10BASE5/2 no mainstream PHYs supported MDROP, it can be said that – after a long break – 10BASE-T1S marks the **renaissance** of this branch of communication technologies.

[(12)] A station loading the bus electrically, the so-called "Babbling-Idiot Failure" problem, etc.

2.3.2 Physical Layer Collision Avoidance (PLCA)

Ethernet mandates the use of CSMA/CD when the PHY is seeking access to the HD (13) media for transmission. The essential part of the "truncated binary exponential backoff algorithm" used by CSMA/CD, introduced in Section 2.1, works as follows in case of, and to resolve, a regular (non-late) collision between multiple transmitters (3):

1. Each of the n transmitters that detected collision interrupts its own transmission;
2. A "jam signal" is emitted by each of the transmitters, to guarantee that none of the non-colliding receivers would falsely accept and process the received frame fragment;
3. Now, each of the transmitters picks its own "uniformly distributed random integer" r from $[0, 2^{\min(i,10)})$ (14) and waits $w = r \cdot slotTimes$ (15);
4. The randomly picked durations – denoted by $w_{1..n}$ for each of the n colliding transmitters – are waited (respectively), upon the expiration of which the next retransmissions are attempted, unless another station already started its transmission in the meantime;
5. If no collision is detected during the repeated attempts, the collision is considered to be resolved, otherwise this algorithm is repeated until each transmitter succeeds or the maximum number of retries – defined by *attemptLimit* (16) – is reached.

What is apparent here are as follows:

- If *attemptLimit* is not defined, then – even under fair conditions – a station attempting transmission may never gain access to the segment, if it gets constantly preempted or collided by other stations at each attempt, and while the probability of this converges to zero at exponential rate, no bound exists on the worst-case frame transmission delay for a HD segment.

  Under unfair conditions this segment's performance may further deteriorate arbitrarily [14];
- If however *attemptLimit* is defined, then – in case of repeated collisions – frames do get silently discarded, thus never reaching their destination.

While non-primitive, real-life communication system are equipped with the necessary mechanisms to detect and recover from frame delivery delays and frame loss, in common use cases the lack of bound on the channel access delay seriously limits the applicability of Ethernet as a networking means with certain use cases.

One of the reasons why Ethernet, especially in HD, achieved little penetration with industrial and automotive use cases was this lack of bound on the channel access delay. This shortcoming was especially prominent in case of functional safety [15], where the

(13) In case of HD Pt2Pt links physical collisions may not occur.

(14) $[\gamma, \zeta)$ denotes the integer interval with the minimum and maximum values of γ and ζ (respectively) that is closed from the left and open from the right, while i denotes the index of the next retransmission attempt.

(15) *slotTime* is defined to be 512 bit times, thus $51.2\,\mu s$, for 10 Mb/s by IEEE Std 802.3-2018 [2] "Table 4–2—MAC parameters".

(16) *attemptLimit* is defined to be 16 for 10 Mb/s by IEEE Std 802.3-2018 [2] "Table 4–2—MAC parameters", but it is often configurable in PHYs.

presence of known (calculable) end-to-end propagation delays greatly simplifies the fundamental analysis of the component and the system.

Due to this, these use cases were often covered by either low-latency/high-end networking technologies, such as Ethernet for Control Automation Technology (EtherCAT), or lower complexity HD MDROP technologies, such as Controller Area Network (CAN), Local Operating Network (LON), FlexRay, etc.).

The relative costs of various phases of Product Change Management (PCM) and the product's Lifecycle Management (LCM) – such as design, development, testing, and maintenance – of complex systems that build on more than one of these diverging branches of technologies are known to be high, and these often require specialized components (gateways with various complexity) to be present in the system. These gateways tend to provide little more than frame and protocol conversion between interconnected networks of different kinds, further increasing total system complexity.

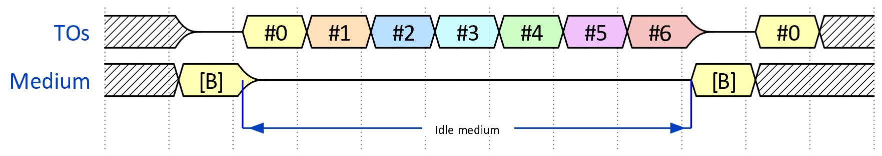
10BASE-T1S comes with a set of features that are meant to bridge this technology gap, while making various features of Ethernet (such as the Ethernet frame (1) itself, the mechanisms that make up Ethernet switching, Virtual LANs (VLANs), etc.) available to implement uniform end-to-end Ethernet communication in highly diverse systems. One of the new features allowing for this to happen is referred to as PLCA [16] and is defined by Clause 148 [17]. While PLCA is specified as an RS, in real-life implementations it is physically incorporated into the PHY "chip", as shown in Figure 2.2.
PLCA is an optional feature of 10BASE-T1S and it offers "physical collision-free" communication within the HD mixing segment, through the following means:

- One of the n stations connected to the mixing segment is configured to be a **coordinator** and is assigned the PLCA Identifier (PLCA-ID) of 0 (zero);
- The remaining $n-1$ stations take the role of being **followers**, each holding a unique PLCA-ID in $[1, n-1]$ (17);
- The coordinator periodically emits a short, but recognizable signal called "BEACON";
- BEACON restarts the PLCA cycle by synchronizing the frame scheduling clocks of all n stations;
- From this point in time on, each station owns a single Transmit Opportunity (TO), in which it may or may not transmit depending on the availability of an egress frame from its MAC:
  - The i^{th} TO may be used only by the single station holding the PLCA-ID $i-1$, thus the coordinator implicitly has the very first TO in each PLCA cycle;
  - When a TO is used by its owner for transmission, all stations – including the transmitter itself – increment their copies of $curID$ (18) and suspend their TO counting until the end of the transmission;

(17) $[\gamma, \zeta]$ denotes the closed integer interval with the minimum and maximum values of γ and ζ (respectively).
(18) $curID$ is defined as "(an) integer variable tracking the ID of the station that currently owns a transmit opportunity" by IEEE Std 802.3cg-2019 [9] "148.4.4.2 PLCA Control variables".

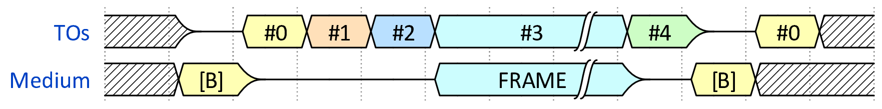
  - When a TO is not used by its owner for transmission, a fixed, but configurable time duration [19] is waited by all stations, after which those increment their copies of $curID$, thus giving the opportunity of transmitting to the next station.

- When – through incrementation – $curID$ would take the value n, the coordinator restarts the PLCA cycle by emitting a new BEACON, which also triggers all copies of $curID$ to be set to 0 (zero), thus restarting the cycle.



Legend:
- #0: Coordinator (single station with zero PLCA-ID)
- #1– #6: 6 followers (stations with non-zero PLCA-ID)
- [B]: BEACON

Figure 2.3: Example timing of a PLCA-based mixing segment with 7 stations



Legend:
- #0: Coordinator (single station with zero PLCA-ID)
- #1– #4: 4 followers (stations with non-zero PLCA-ID)
- FRAME: Actual Ethernet frame with arbitrary length
- [B]: BEACON

Figure 2.4: Example timing of a PLCA-based mixing segment with 5 stations

There are 3 example PLCA scenarios shown, as follows:

- Figure 2.3 shows an example cycle with $n = 7$ and none of the stations transmitting;
- Figure 2.4 displays the case for $n = 5$ and only station holding the PLCA-ID of 3 transmitting;
- Figure 2.5 shows an example for $n = 2$, with both stations transmitting.

[19] The actual duration of a single TO is determined by the timeout of the Clause 148 timer *to_timer* as defined by IEEE Std 802.3cg-2019 [9] "148.4.4.4 Timers". For systems where the MAC and the PHY are not incorporated in the same silicon device, the default and recommended value for *to_timer* is $3.2\,\mu s$. For for integrated solutions however – where worst-case delays at the exposed Media Independent Interface (MII) do not need to be accounted for (such as for example a MAC-PHY) – that is $2.4\,\mu s$. For more details on the difference between these 2 types of manifestations of physical solutions see Section 2.3.3.

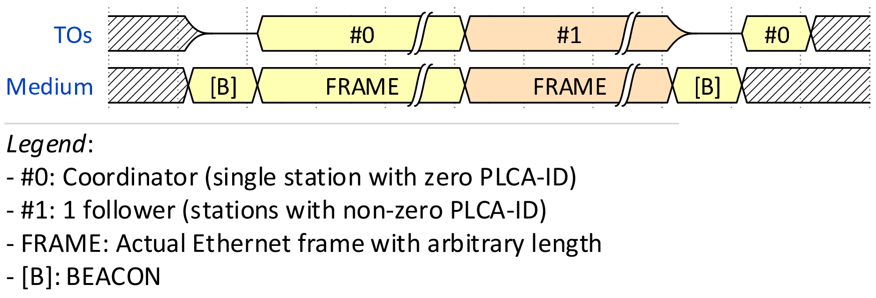

TOs
Medium
#0
#1
#0
[B]
FRAME
FRAME
[B]
Legend:
- #0: Coordinator (single station with zero PLCA-ID)
- #1: 1 follower (stations with non-zero PLCA-ID)
- FRAME: Actual Ethernet frame with arbitrary length
- [B]: BEACON

Figure 2.5: Example timing of a PLCA-based mixing segment with 2 stations

The very task of hiding the apparent contradiction between the philosophy of CSMA/CD-based MAC and a Time-Division Multi(ple) Access (TDMA)-based medium is what PLCA implements, through the following means:

- It does the continuous counting of the PLCA cycles;
- In the ingress direction it does as follows:
  - From the Physical Coding Sublayer (PCS) of the PHY, it receives regular frames, as well as special signals that control the PLCA cycles;
  - Toward the MAC, it forwards received frames.
- In the egress direction it operates as follows:
  - Towards the PCS of the PHY, it schedules egress traffic, while obeying the governance of the current PLCA cycle's TOs;
  - It hides the TDMA-like nature of the medium from the local MAC by a "delay line" internal to the PLCA, and by signaling "virtual collisions", to schedule the MAC's transmission according to the state of the current PLCA cycle.

    Through these means, the MAC is completely unaware of the fact that the medium uses TDMA-like mechanism, and "believes" that in fact it is connected to a segment that uses CSMA/CD.

There are at least the following 3 important observations to make here:

- PLCA is not a traditional TDMA system, as here the duration of each TO is considerably shorter than that of any Ethernet frame. While it is true that even these, very short TOs, do end up causing waste of bandwidth when some stations are idle, the relative waste is negligible even in case of traffic dominated by short frames [20];
- While, for the intuitive first look, it may seem that PLCA naturally implements a priority scheme, giving higher chance of media access to stations holding PLCAs with lower literal values, in fact experiments show [17] that this is not the case and indeed each station has an equal, asymptotically fair share of

[20] Valid Ethernet frames may not be arbitrarily short. Those need to follow the parameter *minFrameSize*, as defined by IEEE Std 802.3-2018 [2] "4.2.3.3,Minimum frame size". The value of *minFrameSize* for 10 Mb/s media is defined to be "512 bits (64 octets)" by IEEE Std 802.3-2018 [2] "Table 4–2—MAC parameters".

the bandwidth. In other words, PLCA is not a priority scheme, and in fact it implements "packet fairness" [18] among all stations on the mixing segment (including the coordinator);

- PLCA does avoid the uncertainty caused by physical collisions on the medium, and it also offers bounded channel access, but in addition to these, experiments show [17] that it also saturates better than HD MDROP segments that rely purely on CSMA/CD for media access.

Even since the relatively recent conclusion of P802.3cg, the practical performance of the PLCA has drawn the attention of the academia, producing results [19] that show the project has succeeded in achieving its objectives in these regards.

2.3.3 Physical manifestations of 10BASE-T1S

Currently there are at least the following 3, vastly different physical manifestations of 10BASE-T1S:

- Standalone **PHY**: A device that incorporates a Clause 147 Physical Medium Attachment Sublayer (PMA) and PCS, as well as a Clause 148 RS (a.k.a. PLCA). The coupling between the Clause 147 and Clause 148 parts is done via the internal MII carrying data nibbles and signaling information [(21)]. Externally, the device exposes a physical MII that is compatible with any Clause 4-compliant MAC.

  Figure 2.6 depicts the structure and interfaces of this architecture.

  The development of engineering samples for silicons based on this architecture have begun much before the final approval of the standard and by now multiple fully conformant products are publicly available [20].

  As explained earlier, the 10BASE-T1S PHY is specified by IEEE Std 802.3cg-2019 [9];

- **MAC-PHY**: A device or solution that integrates an IEEE Std 802.3-2018 [2] Clause 4 MAC and a 10BASE-T1x PHY, exposing a low pin-count Serial Peripheral Interface (SPI) interface to the host Microcontroller (μC) or Microprocessor (μP) with a specialized protocol [21] developed for this purpose. Over this interface the Ethernet frames are fragmented using SPI in both up- and downlink direction, in a way that also allows buffer management, flow control, error detection, monitoring, and configuration.

  Figure 2.7 shows the structure and interfaces of this solution.

  Despite to its relatively recent births, actual products that follow this architecture already exist on the market [22].

  The MAC-PHY is not specified by IEEE, but the Technical Committee 6 (TC6) of the Special Interest Group (SIG) called OPEN Alliance (OA);

- **Analog Front-End (AFE)**: In the context of 10BASE-T1S, AFE is a transceiver device that connects to its host via a 3-pin interface [23]. AFE is analogous to Physical Medium Dependent Sublayer (PMD) in the IEEE terminology, but in this case it incorporates the lower (analog) parts of the Clause 147 PMA.

(21) IEEE Std 802.3cg-2019 [9] "Table 22–1—Permissible encodings of TXD<3:0>, TX_EN, and TX_ER" and "Table 22–2—Permissible encoding of RXD<3:0>, RX_ER, and RX_DV".

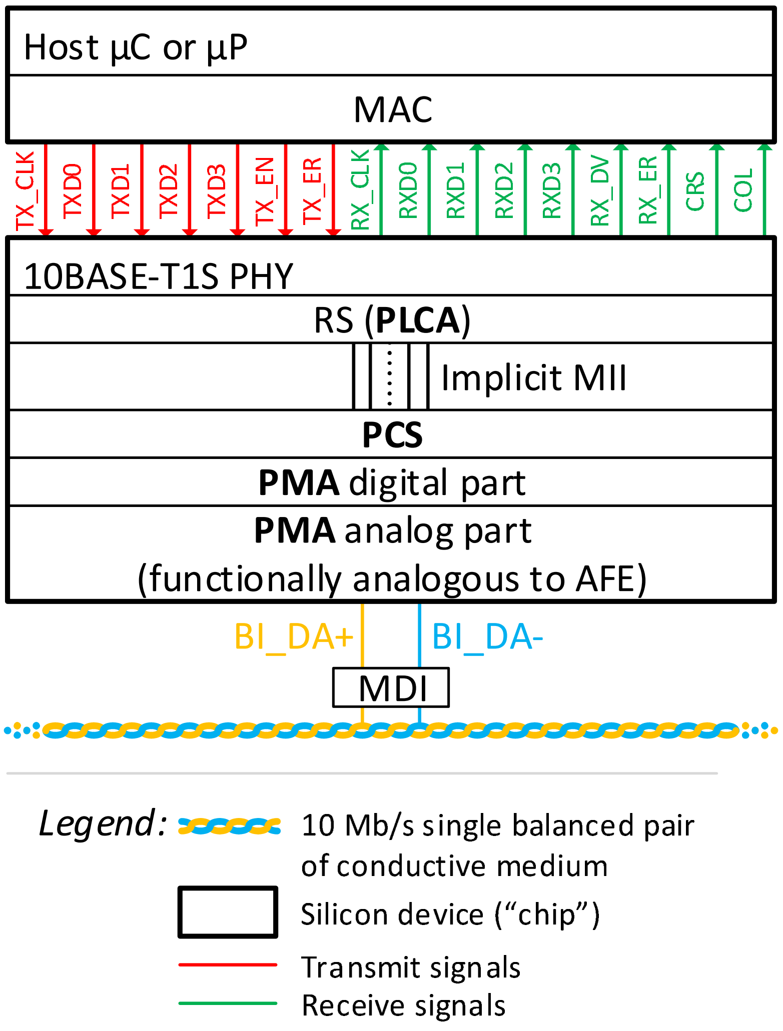


Figure 2.6: Structure and interfaces of a PHY-based 10BASE-T1S system

Using AFE, it becomes possible to decouple the analog and digital parts of the 10BASE-T1S, this way enabling – even multiple instances of – the digital part of the 10BASE-T1S to be integrated into the host μC or μP. Such arrangement is very similar to how existing industrial and automotive networks, such as CAN, LON, or FlexRay, are layered and architected.

Figure 2.8 depicts the structure and interfaces of this architecture.

Despite the fact that this architecture has been invented most recently, some products supporting also this concept are already commercially available [24].

The AFE is not specified by IEEE, but the Technical Committee 14 (TC14) of OA.

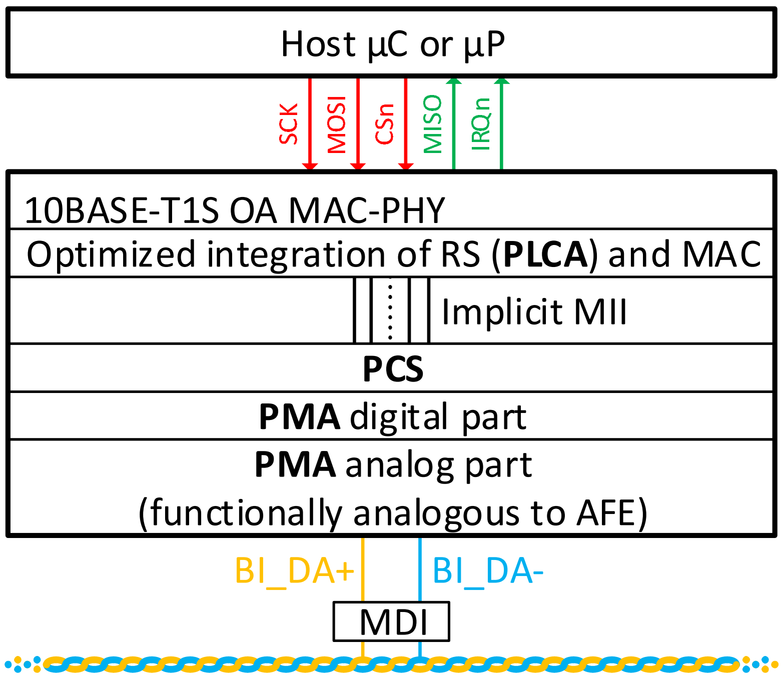


Figure 2.7: Structure and interfaces of a MAC-PHY-based 10BASE-T1S system

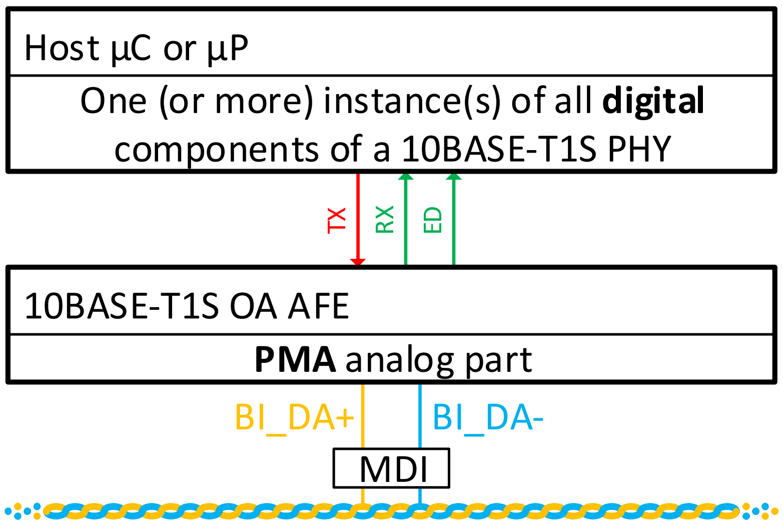

Host µC or µP
One (or more) instance(s) of all digital components of a 10BASE-T1S PHY
TX
RX
ED
10BASE-T1S OA AFE
PMA analog part
BI_DA+
BI_DA-
MDI

Figure 2.8: Structure and interfaces of an AFE-based 10BASE-T1S system

2.3.4 Applicability of 10BASE-T1S

In its mature form, 10BASE-T1S is expected to be the missing link in the technology, that allows majority of industrial, automotive, backplane, etc. use cases to become a member of the Ethernet eco-system. The following non-exhaustive list shows typical use cases from all these segments of the technology:

- **Industrial segment**: Alternative technology for CAN, LON, SPI, Inter-Integrated Circuit (I^2C), and proprietary RS-485-based networks;
- **Automotive segment**: Competition for FlexRay, CAN XL [25], and other proprietary in-car networks covering all end-to-end communication needs that have lower bandwidth demand (thus excluding autonomous driving).

  Due to its supported maximum reach, 10BASE-T1S is applicable in case of networks in trucks, lorries, trams, trains, etc.;
- **Backplane segment**: Uniform networking technology allowing intra-system communication between high-complexity components of bridges, routers, servers, etc. While at its formation P802.3cg has defined 10BASE-T1S to target **twisted** single-pair media, the specification has later been relaxed by removing the requirement for twisting (6), to allow this segment of use cases to be covered, as in Printed Circuit Boards (PCBs) twisting of copper traces is a complex problem to solve [26].

As explained in Section 2.3.2, due to its bounded channel-access time, 10BASE-T1S is expected to be able to go beyond general-purpose communication, and be the preferred choice for extended uses cases, for example those of functional safety.

2.4 The IEEE Project 802.3da

In one hand – as explained in Section 2.3.2 and Section 2.3.4 – P802.3cg has succeeded in reviving MDROP mode of operation for Ethernet while offering bounded channel access time and packet-fairness. On the other hand however, 10BASE-T1S was not freely applicable in so-called "non-engineered" networks, that would also need to withstand industrial and automotive noise conditions [27], and where Plug-and-Play (P&P) is of the essence [28].

For these reasons, following the conclusion of P802.3cg, the extended requirements and the cooperation of experts gave birth to the IEEE Project 802.3da [29] (P802.3da) to collect, evaluate, filter, and ultimately meet these extended demands. P802.3da started to operate under its own PAR [30] and along its own objectives [31] aligned with the effort of enhancing 10BASE-T1S with MDROP.

The objectives of P802.3da have been set very specifically to build on what 10BASE-T1S has already achieved, while improving it specifically along a set of requirements that would broaden the usability and improve the interoperability of the successor of 10BASE-T1S, called 10BASE-T1M.
The project attempts to meet these requirements by making 10BASE-T1M support the following features:

- Extension to the minimum guaranteed reach of, and station count on the mixing segment;
- Implementation of a common PLCA-ID allocation that makes the automatic boot-strapping of a mixing segment P&P and vendor-independent;
- Support for node addition and removal of stations to/from a possibly powered mixing segment (a.k.a. "hot-plugging"), including proper handling of churn, and in a way that powers the segments iff PDs are present;

- Addition of optional support for Time Synchronization Service Interface (TSSI);
- Selection of a single MDI connector, yet again in support of P&P;
- Improvement of the Energy Efficient Ethernet (EEE) characteristics of the PHY;
- "Support operation in the noise environments for building, industrial, and transportation applications" while maintaining all of the above;
- Achieve all these while remaining interoperable with 10BASE-T1S also in the MDROP mode of operation.

At the writing of the dissertation, P802.3da is in its active (early) phase.

2.5 Research motivations

Throughout the entire research presented in this dissertation, the motivation has consistently been the enhancement of the noise resilience of the PHY and improving its other performance metrics, especially under industrial conditions.

2.5.1 During P802.3cg

The research has been started by making a set of initial contributions in P802.3cg [13, 32, 33] defining the industrial environments of interest. Later the horizon of the contribution has been widened by taking editorial role for Clause 147.

The task of an editor in IEEE 802 is to collect comments [34] on the continuously developing drafts of the standard, and – in coordination and cooperation with the contributing experts of the field – propose impartial resolution for each of the comments. The WG then holds a face-to-face meeting with open participation where a final resolution is accepted in a transparent, rule-based manner, using proven management principles (22), under the direction of the clause- or the chief editor. Finally, the editor executes the accepted resolution against the draft version of the standard. This role therefore allows an insight not only to the process, but also gaining an understanding of the technical challenges the project faces.

During P802.3cg the research motivation has been to understand how good the choice of 10BASE-T1S **frame preamble** was in technical terms, including whether a constant pattern with better properties could be proposed, with the intention of improving noise resilience and reliability of the Beginning of Frame (BoF) detection, as well as optionally enabling the implementation of new features, such as Operations, Administration, and Maintenance (OAM) and Out of Band (OoB) signaling, etc. This phase of the research yielded new results. Part of these have been presented earlier [35], while the entirety of the results are presented in Chapter 3 of the dissertation.

Throughout the research and development done in P802.3cg additional contributions have been made with the intention of improving the technology through fixing shortcomings [36, 37] in, as well as adding new features to it [38].

2.5.2 During P802.3da

Built on an improved frame preamble, the planned continuation of the research was to add noise resilience also to the payload of the frame, by proposing a Forward Error

(22) Most typically "Robert's Rules of Order" are applied.

Correction (FEC) method for the payload. While it would have been very natural to combine the preamble- and FEC-originated themes of the research to implement burst noise immunity for the entire Ethernet frame – as explained in Section 2.4 – in the meantime P802.3da has been started along a new set of 11 objectives, that made the research adjust its direction.
Out of all the objectives of the P802.3da, the following have strong relevance with respect to how an FEC can – and indeed shall – be constructed:

- **Objective #1** [(23)] in combination with **objective #2** [(24)]: It has been shown [39] that to maintain acceptable Bit Error Rate (BER) at MAC/PLS with the increased reach demanded by P802.3da may require means not currently used in 10BASE-T1S. One of solution candidates considered was Decision Feedback Equalization (DFE), however DFE is known [40] to produce bursts of errors;

- **Objective #8** [(25)]: While noise immunity is of concern for all communication systems and methods, it has not been addressed explicitly by P802.3cg. Objective #8 of P802.3da however is strongly driven by industrial and automotive use cases where devices and systems need to withstand harsh burst noise stimuli, strongly affecting coding, as well as component design [41];

- **Objective #11** [(26)]: This feature – a.k.a. “hot-plugging” – is alone a strong and hard to characterize source of burst error. This problem is made even more challenging to solve [42] by P802.3da allowing addition and removal of **set of stations** while operational power is also supplied over the mixing segment;

- **Objective #4** [(27)]: this objective mandated the PHY to remain **interoperable** with the 10BASE-T1S over MDROP as defined by Clause 147 and Clause 148.

  While an explicit definition for the term “interoperable” has not been given in P802.3da, this research builds on the assumption that any new feature that is **backward-compatible** would automatically meet the interoperability requirements.

According to this, burst noise resilience for the payload of the Ethernet frame could be established by designing a backward-compatible FEC that is capable of correcting burst errors, and optionally also burst erasures. Parts of the results produced by this theme of the research have earlier been made available [43, 44, 45], along with the notable simulations and verification methods [46, 47], while the entirety of the latest results are presented in Chapter 4 of the dissertation.

Irrespective of the interpretation of the interoperability objective built on in the dissertation, active discussions exploring other interpretations as well are ongoing as we speak [48, 49, 50], including the possibility of entirely discarding objective #4.

During the research and development done in P802.3da additional contributions have also been made that aimed at improving resilience aspects of the technology outside of the noise-domain [51].

[(23)] “Define performance characteristics of a mixing segment for 10 Mb/s multidrop single balanced pair networks supporting up to at least 16 nodes, for up to at least 50m reach”.
[(24)] “Maintain a bit error ratio (BER) at the MAC/PLS service interface of less than or equal to 10^{-10} on the new mixing segment”.
[(25)] “Support operation in the noise environments for building, industrial, and transportation applications”.
[(26)] “Support addition and removal of a node or set of nodes to a continuously operating powered mixing segment”.
[(27)] “Support interoperability with Clause 147 multidrop”.

For the sake of completeness, it is also worth mentioning that the noise immunity- and reach extension-related efforts have seen contributions [52, 53] already during P802.3cg, thus long prior to the establishment of P802.3da. These were to build on increasing the **transmit voltage** of the PHY, thus these are crucially different than an FEC-based direction. A higher transmit voltage, may increase Signal-to-Noise Ratio (SNR) in case of external noise, and it is one of the trivial means to counter Insertion Loss (IL), however it not only has clear disadvantages as well – such as increased Intersymbol Interference (ISI) and the increased level of Electromagnetic Interference (EMI) – it has been firmly expressed by the silicon vendors that the technology that could make 10BASE-T1S implementations economically feasible would not be capable of outputting an increased voltage necessary for the solution to be implemented using the most commonly available device supply voltage levels.

Chapter 3 On the 10BASE-T1S preamble for multidrop

3.1 Introduction

Work presented in this chapter focuses on the background, history, and evolution of the Ethernet frame preamble chosen for 10BASE-T1S, then it proposes new preamble candidates that enable the implementation of additional communication functions. These efforts are rooted in simulations done in the analog domain, the technical details of which are to be explained in Section A.1.

A good preamble allows simple and error-resilient detection of the beginning of a Physical Layer Device (PHY) frame, while also providing reliable hints for the underlying system with respect to the synchronization of the clocks of essential functions in the receiver. A pair of good preambles also enable different types of frames to coexist on the same (shared) medium, while maintaining a low probability of interference between these.

As the development process that led to the selection of the current constant preamble pattern used today by 10BASE-T1S has seen grossly different competing ideas, as well as contradicting results, the initial research motivation was to understand the content, validity and correctness of the presented techniques. Based on the results collected during this initial phase, research also intended to propose new preamble(s) that would be at least as good as the one currently in use.

3.1.1 About the preamble

Ethernet is a frame-based communication method, where information is transferred as blocks (a.k.a. "chunks") of data, referred to as **frame**s (a.k.a. "packets"). An Ethernet frame has well defined components described by the standard [(28)]. As opposed to the "ideal" frame, when transmitted over the 10BASE-T1S medium, the Ethernet frame features additional and modified components in the beginning and end of each frame, as shown in Figure 3.1.
As visible, there are at least the following two possible interpretations of the term *preamble*:

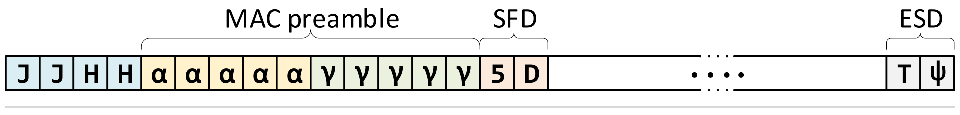
- The 16 4B symbols (a.k.a. "nibbles") sent and received by the Media Access Control (MAC) at Media Access Control/Physical Signaling (MAC/PLS): The sequence of 5B-mapped symbols that are based on these are referred to as "MAC preamble" in Figure 3.1. When on the medium, the initial $4 + 5 = 9$ of these 16 5B symbols get overridden and then restored by the transmitting and receiving Physical Coding Sublayers (PCSs) of the acting PHYs (respectively);

- The constant sequence of 4 5B symbols sent and received by the PCSs of the PHYs: this is referred to as 10BASE-T1S frame preamble in Figure 3.1.

In subsequent parts of the dissertation, the term *preamble* will refer to the 10BASE-T1S frame preamble.

In short, one of the main roles of the preamble is to permit reliable and error-resilient detection of the beginning of the frame. The actual values (signal pattern) used as preamble are (is) typically PHY-specific and may or may not consist of symbols that are used to build the subsequent parts of the frame.

[(28)] IEEE Std 802.3-2018 [2] "Figure 4–5—Frame with carrier extension".



Legend:
- **J** and **H**: The constant 5B symbols 'J' (binary 11000) and 'H' (binary 00100)
- **α**: 5B symbols the 4B symbol mapping of which is used for the synchronization of the multiplicative scrambler (partially or fully)
- **γ**: 5B symbols typically conveying the constant 5B symbols '5' (binary 01011)
- **5** and **D**: The constant 5B symbols '5' (binary 01011) and 'D' (binary 11011) encoding the SFD for the local MAC
- **T**: The constant 5B symbols 'T' (binary 01101)
- **ψ**: For regular frames the constant 5B symbols 'R' (binary 00111) For errored frames the constant 5B symbols 'K' (binary 10001)

🟦 **10BASE-T1S frame preamble** 5B symbol, *generated by the transmitting PHY*

🟨 Scrambler lock 5B symbol, *generated by the transmitting PHY*

🟩 Remaining (verbatim) part of the MAC preamble, *generated by the transmitting MAC*

🟥 Start of Frame Delimiter (SFD), *generated by the transmitting MAC*

⬜ 5B symbol with valid 4B symbol mapping carrying the payload of the frame, including DADR, SADR, length/type, user data and FCS, *generated by the transmitting MAC*

⬜ 10BASE-T1S frame postamble 5B symbol, *generated by the transmitting PHY*

Figure 3.1: The Ethernet frame on 10BASE-T1S

Preambles are often used in frame-based (a.k.a. packet-based) communication systems and are known to be well-recognizable with correlators in the presence of *frequency and phase offset* [54] and/or *noise* in case of both wireline and wireless networking [55]. To maintain grounds for comparison of results however, we will utilize the model and method agreed withing IEEE Project 802.3cg [4] (P802.3cg) for the reasons explained in Section 3.2, but the aim of our work is aligned with the general requirements on these systems, which is to minimize both the false negatives and false positives – or *false alarm* [56] – at the decisions on match.

3.1.2 The roles of the preamble

Section 3.1.1 has highlighted that formally the main role of the preamble is to allow reliable detection of the beginning of the PHY frame, however its practical roles go beyond those, as summarized by the following list:

1. **Detection** of the beginning of the PHY frame (as explained earlier): This has been introduced by "147.3.3.1 PCS Receive overview", then formally defined by the Finite State Machine (FSM) following "Figure 147–7—PCS Receive state diagram, part a" of IEEE Std 802.3cg-2019 [9];

2. **Clock recovery for** the Differential Manchester Encoding (DME) line code **bits**: This has been introduced and normatively defined by "147.4.3 PMA Receive function" of IEEE Std 802.3cg-2019 [9];

3. **Clock recovery for** the 5B **symbols**: This has also been defined by "147.4.3 PMA Receive function" of IEEE Std 802.3cg-2019 [9].

A naïve approach is to implement the preamble detection literally as specified by these three sources in the standard, however the performance of such a solution would be prone to mischaracterization for a number of reasons. As an example, when a receiver's Physical Medium Attachment Sublayer's Receive Function (PMA_RX) is not locked, and its Physical Coding Sublayer's Receive Function (PCS_RX) is in `WAIT_SYNC` state, if quasi-random digital samples arrive at the PMA_RX, then – due to the simple rules of DME – lock type #2 is very likely to be achieved after a few bit times. This lock is followed by lock type #3, again with a high probability, simply because most of the encodable 32 5B symbols are in use by the PHY (the details of the 5B symbols will later be shown in Table 4.2), thus the majority of existing 5B symbols would be recognized. For these reasons, in noisy conditions, this process would oftentimes lead to false positives. Such chain of events would cause PMA_RX receive quasi-random 5B symbols, which may not only cause the preamble detection itself to signal a false positive, but may also cause a loss of synchronization between the Physical Layer Collision Avoidance (PLCA) cycles of all the stations on the mixing segment, therefore making the maintenance of bounded channel access time more problematic.

As explained above, such a simple implementation is prone to false positives when it comes to preamble detection, however it is also susceptible to false negatives, simply because a very small set of samples affected by external noise or Intersymbol Interference (ISI) may also momentarily make PMA_RX lose lock type #2, lock type #3, or both, and either of those event would prevent seamless reception of the frame.

P802.3cg considered such weaknesses of a naïve implementation, and defined the preamble detection separately from the rest of the PMA_RX and PCS_RX functions. This enables implementations that solve this problem using **correlators** [57]. The correlator-based approach uses an Aperiodic Autocorrelation (AAC) function that – in this case – is used to quantitatively express the difference between an **ideal** waveform and **actual** waveform received by the PMA_RX.
While even a simple correlator needs additional silicon gates that would otherwise not be necessary for a naïve implementation, in return it does offer the following advantages:

- Avoiding **false negatives** and dealing with distorted input waveforms: Instead of providing a **binary** "match/no-match" type of value, AAC expresses level of similarity in the form of a **real value**, where the closer the – typically distorted – actual input signal is to the ideal waveform, the higher the AAC value it will get. This not only means that noise affecting the input signal will not automatically make the detection fail, but the acceptance criteria of the detector may even be adjusted according to the live state – for example the Signal Quality Indicator (SQI) value for – of the segment;

- Locking on the bit and the symbol **clocks**: With preambles that have "good" AAC properties, the peak values of the AAC function's output may be used to achieve both type of locks (lock type #2 and lock type #3) at the same time. The details of this is described in more details in Section 3.2.

3.2 The goodness metrics of the preamble

As explained in Section 2.2, two of the main design motivations behind 10BASE-T1S were maintaining low complexity and being capable of meeting industrial and automotive demands. In the design of the preamble however, there is the following trade-off between these:

- In one hand, low complexity requirement must support a design that is friendly to implementations without high-resolution and -speed Analog-to-Digital Converters (ADCs). As an example a 10BASE-T1S relying purely on threshold decoding and **binary correlators** [57] should be implementable based on the standard;
- On the other hand little-to-no compromise shall be made with regards to the reliability of the preamble detection, even on a medium that is influenced by noise.

To meet these demands, multiple experts in P802.3cg have done extensive research to first find a well defined **model** that produces repeatable and comparable results, then to use this consensus-based model to evaluate preamble pattern **candidates**. This process yielded the following agreements and conclusions:

- A single **channel model** would serve as a basis to compare upcoming results, as further described in Section 3.2.1;
- To make the preamble well **recognizable**, the pattern candidates are passed through the channel model and – instead of relying directly on the Golay Merit Factor [58] – their AAC properties (within themselves) are to be considered, as further explained in Section 3.3.3.A;
- To make the preamble well **distinguishable** from other constant patterns that may appear on the medium under the same conditions, the Aperiodic Cross Correlation (ACC) properties of the undistorted candidates would be used, without considering the affect of the channel model, as further described in Section 3.3.3.B.

The channel model used will be introduced in Section 3.2.1, while the topics related to AAC and ACC are covered in Section 3.2.2.

3.2.1 Channel model for preamble evaluation

As explained in Section 3.2 AAC has been used to evaluate goodness of preamble candidates. In Section 3.2.2.A it is shown that while in general AAC means calculating the similarity of a waveform with itself using different shift values, in the context of the preamble evaluation this technique is slightly modified, and – instead calculating the correlation between the signal and itself – the method verifies the correlation between the ideal signal and its version distorted by the complete channel model, which is shown in Figure 3.2.
The explanation for the functional elements in Figure 3.2 is as follows, while the arrows labeled as $s_1..s_6$ represent the signals travelling between these elements:

1. An ideal PHY core that emits the perfect, undistorted analog waveform that is being evaluated.

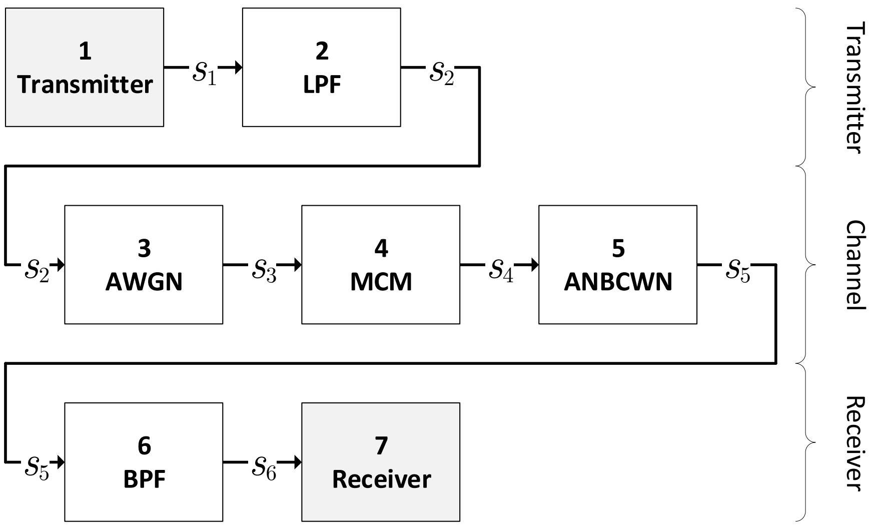


Figure 3.2: Channel model used in P802.3cg to evaluate preamble pattern candidates

Here, the DC-balanced signals were following the nominal peak-to-peak swing of 1 V (0.5 V_{RMS}) ±20% defined by the standard [29];

2. Low-Pass Filter (LPF) attached to the transmitter station, producing the waveform that is entering the channel.

   For this, a 2nd-order digital Butterworth LPF with the corner frequency of 30 MHz was selected;

3. An Additive White Gaussian Noise (AWGN) signal coupled to the channel while being transmitted over the medium.

   This noise source had the Carrier-to-Interference Ratio (CIR) of -30 dBc [30];

4. Max Cable Model (MCM) representing the signal distortion imposed by a maximum configuration mixing segment, with respect to station count, cable length and capacitance as described in more details in Section 3.2.1.A;

5. A set of Added Narrow-Band Continuous Wave Noise (ANBCWN) signals coupled to the channel while being transmitted over the medium.

   During the simulations, altogether $8(2(30-1)+1) = 472$ different **continuous sine-waves** with the swing of 0.5 V ($\approx$ 0.177 V_{RMS}) covering the frequency range of $[1, 30]$ Mhz (in stepping of 500 kHz) and phase range of $[0, \frac{7}{4}]\,\pi$ (in stepping of $\frac{\pi}{4}$) were generated and added as disturbance to the channel. This selection of parameters resulted in $\approx$ 9 dB of Signal-to-Noise Ratio (SNR);

6. Band-Pass Filter (BPF) attached to the receiver station, affecting the signal exiting the channel.

   For this, a 2nd-order digital Butterworth BPF with the corner frequencies of 1 MHz and 30 MHz was selected.

[29] IEEE Std 802.3cg-2019 [9] “147.5.4.1 Transmitter output voltage”.

[30] dBc stands for “decibels relative to the carrier”, and it expresses power ratio of a signal relative to the carrier signal.

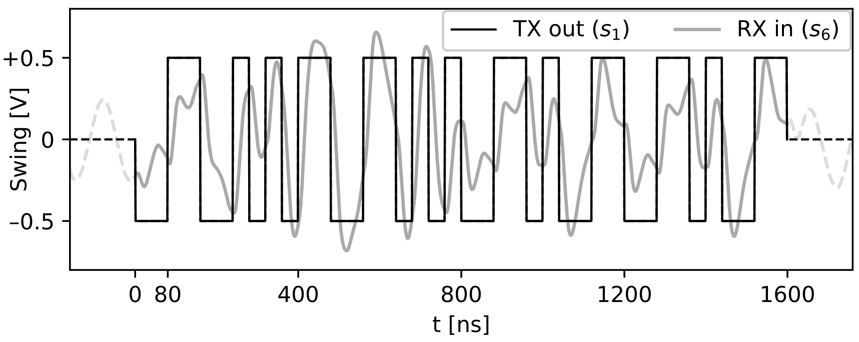


Figure 3.3: A possible pair of ideal signal and its distorted counterpart on the two sides of the channel model

An example of what total distortion on such a model may impose on the ideal preamble is shown in Figure 3.3, in which the signal labeled “TX out” corresponds to s_1 in Figure 3.2, and the signal labeled “RX in” represents s_6 in Figure 3.2.

3.2.1.A Max Cable Model (MCM)

The 3 functional elements listed in Section 3.2.1 that belong to the channel are #3, #4, and #5, out of which #3 and #5 represent external noise coupled to the medium at the transmitter and the receiver (respectively), while #5 represents the assumed worst-case electrical effect of the channel itself, including cabling, connectors and the Media Dependent Interfaces (MDIs) of the stations. The following considerations apply when simulating the total effect of MCM:

- **Insertion Loss (IL)** [(31)]: The maximum (limit) values permitted by the standard [(32)] for the frequency range of $[0.3, 40]$ Mhz are applied, according to Equation (3.1);

$$\text{Insertion\_loss}(f) \leq \left\{ \begin{array}{ll} 1.0 + \frac{1.6(f-1)}{9} & \text{for } 0.3 \leq f < 10 \\ 2.6 + \frac{2.3(f-10)}{23} & \text{for } 10 \leq f < 33 \\ 4.9 + \frac{2.3(f-33)}{33} & \text{for } 33 \leq f \leq 40 \end{array} \right\} \text{[dB]} \tag{3.1}$$

where f is the frequency in MHz, and $\{f \in \mathbb{R} \mid 0.3 \leq f \leq 40\}$

- **Return Loss (RL)** [(33)]: The standard does define limits for RL [(34)] as shown by Equation (3.2), however its effect is not part of MCM, as Multidrop (MDROP) is always half duplex;

(31) Insertion Loss (IL) is defined as $\text{IL} = -20\log_{10}(|S_{21}|)\,\text{dB}]$, where S_{21} is one of the scattering parameters of the channel.
(32) IEEE Std 802.3cg-2019 [9] “147.7.1 Insertion loss”.
(33) Return Loss (RL) is defined as $\text{RL} = -20\log_{10}(|S_{11}|)\,\text{dB}$, where S_{11} is one of the scattering parameters of the channel.
(34) IEEE Std 802.3cg-2019 [9] “147.7.2 Return loss”.

$$\text{Return\_loss}(f) \geq \begin{cases} 14 & \text{for } 0.3 \leq f < 10 \\ 14 - 10\log_{10}(\frac{f}{10}) & \text{for } 10 \leq f \leq 40 \end{cases} \text{[dB]} \tag{3.2}$$

where f is the frequency in MHz, and $\{f \in \mathbb{R} \mid 0.3 \leq f \leq 40\}$

- Maximum total **capacitance**: Clause 147 sets no constraints on the possible geometry/arrangement of stations over the mixing segment, thus the number of possible combinations is not limited. Simulations run during the development phase of P802.3cg show [59, 60] that the worst-case scenario is likely when all capacitance accumulates (lumps) at the far-end of the pair of conductors (called "clumped arrangement"). The research presented in the dissertation relies on this assumption of those simulations, which suggests that the effect of the maximum total capacitance to the signal is an additional 0.85 dB (35) of loss above IL.

3.2.1.B Absolute worst-case channel model

It is worth pointing out that this model may not be best fit for finding the preamble with the highest possible performance under absolute worst conditions, as it ignores the following factors:

- **Droop**: Clause 147 has a permissive droop specification (36);
- Permitted variation of the **swing**: Clause 147 allows ±20% (29) variation in the transmitter voltage;
- **Insertion Loss (IL)**: The channel model applies the IL according to the limits shown in Equation (3.1), however – from the receiver's perspective – the worst-case signal attenuation values would be as follows:
  - Below and above band (37): 0 dB;
  - In-band (38): as per Equation (3.1).

Despite these differences, our research used the commonly agreed channel model described earlier in Section 3.2.1, as this provided the only basis for comparison of results.

3.2.2 Discrete correlators for time series

For the sake of simplicity, we define the discrete ACC as shown in Equation (3.3) [61].

$$\text{ACC}_{s_a,s_b}(\tau) = \begin{cases} \sum_{i=1}^{L-\tau} s_a(i+\tau)\cdot s_b(i) & \text{if } 0 \leq \tau < L; \\ \text{ACC}_{s_a,s_b}(-\tau) & \text{if } \tau < 0; \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

The notations in Equation (3.3) go as follows:

- The discrete real-valued function $\text{ACC}_{s_a,s_b}(\tau)$ produces a single real value that expresses the ACC between the sample arrays (time series) s_a and s_b with a shift;

(35) $3.7 - \text{Insertion\_loss}(12.5) = 0.85\,\text{dB}$

(36) IEEE Std 802.3cg-2019 [9] "147.5.4.2 Transmitter output droop".

(37) $[0.3, 6.25)$ Mhz and $(12.5, 40]$ Mhz (respectively).

(38) $[6.25, 12.5]$ Mhz.

- τ is this shift value – expressed in number of samples – present between (applied to) s_a and s_b;
- L denotes the length of – a.k.a. number of samples in – s_a and s_b, thus length of s_a and s_b should either equal, or the shorter is to be padded with zero values (typically symmetrically).

3.2.2.A AAC for the 10BASE-T1S preamble

AAC is also a real-valued discrete function, the definition of which in the general context is shown by Equation (3.4).

$$\mathrm{AAC}_{s_a}(\tau) = \mathrm{ACC}_{s_a,s_a}(\tau) \tag{3.4}$$

In the context of the research on the preamble candidates however, AAC is defined as shown by Equation (3.5). Here s_1 and s_6 follow the definition given in Section 3.2.1.

$$\mathrm{AAC}_{s}(\tau) = \mathrm{ACC}_{s_1,s_6}(\tau) \tag{3.5}$$

From these 3 formulæ, it is visible, that if AAC is calculated for each meaningful value of τ, the process generates a series of $2L-1$ (τ, AAC) pairs for a waveform of L samples. Plotting and analysing these AAC values against their τ counterparts will be the basis for evaluating the preamble waveform candidates, presented in subsequent sections.

3.2.2.B ACC for the 10BASE-T1S preamble

Analogously to the reference research, the ACC-based evaluation of the preamble omitted the channel model introduced in Section 3.2.1. Instead of applying the channel model, the ideal forms of the two waveforms in question are calculated directly (without transformation) using Equation (3.3).

3.3 Evolution of the 10BASE-T1S preamble

3.3.1 Historical outline

During its research and development process to find the pattern to be used by 10BASE-T1S as preamble, P802.3cg initially preferred the constant 4-element long 5B symbol sequence of `JJJK`.

The presentation proposing `JJJK` [62] has shortly been followed by a competing idea and presentations [63, 64] that proposed using two constant Golay Complementary Sequences (GCSs) (39) [58] to construct a considerably longer preamble we will refer to as GCS-based Preamble (GCSP) (40).

The latter presentation [64] has shown that using the agreed channel model presented in Section 3.2.1, Ga32 and Gb32 outperformed `JJJK` and `NNNN` (41) (respectively)

(39) Namely Ga32 and Gb32, where Ga32 refers to the ternary sequence of `[+,-,+,+,-,+,+,+,+,-,+,+,-,+,+,+,-,+,-,-,-,+,+,+,+,-,+,+,+,-,-,-]`, Gb32 is that of `[-,-,-,+,+,+,-,+,-,-,-,+,+,+,-,+,+,+,+,-,+,+,-,+,-,-,-,+,-,-,+,-]`.

(40) GCSP=Ga32|Z(32)|Gb32|Z(16), where Z(n) denotes n consecutive zero-valued ternary symbols, and | represents the concatenation operation.

(41) At the time of that simulation `NNNN` was being considered as the BEACON signal for PLCA.

regarding their AAC properties. In addition to this, GCSP would have been suitable for Harness Defect Detection (HDD) as well.

As a response to this challenge the researchers who originally proposed `JJJK` conducted extensive comparative research, and came back with the new preamble candidate of `JJHH` replacing `JJJK`. Their new presentation [65] has shown that `JJHH` outperforms both Ga32 and Gb32. After considerable debate the project settled on using `JJHH`, which is also shown in Figure 3.1, thus `JJHH` became the preamble of 10BASE-T1S.

Interestingly, one of the initial presentations [63] suggested using Golay Merit Factor [58] as a basis for comparison of candidates, this method has later been replaced by the method, presented in Section 3.3.3.A.

3.3.2 Quest to find a better preamble

According to the explanation in Section 2.5, understanding, verifying and enhancing the goodness of `JJHH` was the motivation of the theme of the research presented in this chapter. This research has started off from the conclusion described in Section 3.3.1 and it had two distinct phases:

1. **Reproducing results** shown in the key presentations [64, 65] leading to `JJHH` being selected. The simulations serving as a basis for these presentations had proprietary details that the parties (industry-leading silicon vendors) did not fully disclose. For example, some of the key details of the channel model described in Section 3.2.1 were not fully defined. To be able to do a fair and quantitative comparison, research required earlier results to be reproduced;

2. Based on the output of `#1`, the final goal was attempting to find **better preamble** than `JJHH`.

3.3.3 The simulation environment

The simulation environment used to evaluate the performance (goodness) of a preamble waveform candidate consisted of the following components and steps:

- Two separate formula for AAC and ACC, as explained in Section 3.2.2.A and Section 3.2.2.B (respectively);

- To evaluate the **recognizability** of a single waveforms, ACC was used. In this case ideal forms of the two waveforms and various distorted versions of it were fed to Equation (3.3) with all meaningful values of τ. Section 3.3.3.A includes further details on this process;

- To weigh the **distinguishability** of a pair of distinct waveforms, ACC was used. In this case channel model was bypassed and the ideal forms of the two waveforms to cross-correlate were fed to Equation (3.3), yet again with all meaningful values of τ. Section 3.3.3.B outlines more details on this process.

3.3.3.A Goodness of recognizability

The simulation that established means to measure the goodness of recognizability of a preamble consists of the following steps:

1. The ideal waveform (time series) of the preamble s to be evaluated is generated as an array of L real-valued samples for the simulation environment (42), serving as a single s_1;

2. The samples representing s_1 are fed to the channel simulator that acts according to the model described in Section 3.2.1, the output of which is again an array of L real-valued samples representing the distorted version of the ideal signal, referred to as s_6. As the actual content of s_6 depends on which one of the 472 ANBCWN is used by the channel model, indeed this step is executed 472 consecutive times with different ANBCWN for each run, producing 472 different s_6 arrays, subsequently referred to as $s_{6,n}$, where n is in $[1, 472]$;

3. Equation (3.5) is used to calculate AAC series for each $s_{6,n}$ and for all meaningful values of τ, producing $\mathrm{AAC}_{s,n}$, each holding $2L - 1$ AAC values.

   Figure 3.4 shows an example of what all $\mathrm{AAC}_{s,n}$ plotted in parallel may look like for `JJJK`. It is worth mentioning that the unit of the horizontal axis is T3, which is a parameter defined by P802.3cg to characterize DME timing, and the nominal value of which is 40 ns (43);

4. Each of the $\mathrm{AAC}_{s,n}$ series are searched for 3 notable points, as follows:
   - $\ell 1_n$: The highest valued **positive peak** (a.k.a. the **local maximum**) in $\mathrm{AAC}_{s,n}$. The location (τ), and its immediate neighborhood, where $\mathrm{AAC}_{s,n}(\tau)$ takes the value of $\ell 1_n$ is subsequently referred to as the **main lobe** of $\mathrm{AAC}_{s,n}$. If $\mathrm{AAC}_{s,n}$ has no single highest valued peak, the simulation is aborted with an error;
   - $\ell 2{-}_n$: The lowest valued **negative peak(s)** (in other words the **local minimum/minima**) in $\mathrm{AAC}_{s,n}$. The location(s) (τ or τs), and its/their immediate neighborhood(s), where $\mathrm{AAC}_{s,n}(\tau)$ takes the value of $\ell 2{-}_n$ is/are subsequently referred to as the **secondary negative lobe(s)** of that series. If $|\ell 2{-}_n| \geq |\ell 1_n|$ for any value of n, the simulation is aborted with an error;
   - $\ell 2{+}_n$: The 2nd highest valued **positive peak(s)** in $\mathrm{AAC}_{s,n}$ is/are identified as well. The location(s) (τ or τs), and its/their immediate neighborhood(s), where $\mathrm{AAC}_{s,n}(\tau)$ takes the value of $\ell 2{+}_n$ is/are subsequently referred to as the **secondary positive lobe(s)** of that series.

5. Based on this, the following definitions are made:
   - $\ell 1$ is defined as the *minimum absolute value* among all $\ell 1_n$;
   - $\ell 2-$ is defined as the *minimum negative value* among all $\ell 2{-}_n$;
   - $\ell 2+$ is defined as the *maximum positive value* among all $\ell 2{+}_n$.

6. Finally, all the 472 $\mathrm{AAC}_{s,n}$ series are normalized around $\ell 1$.

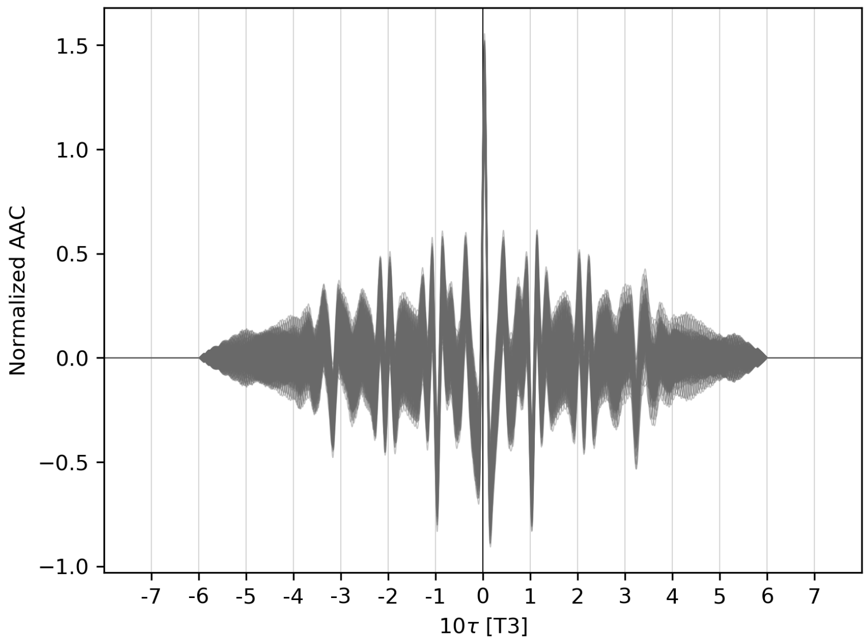


Figure 3.4: Aperiodic Autocorrelation (AAC) of JJJK

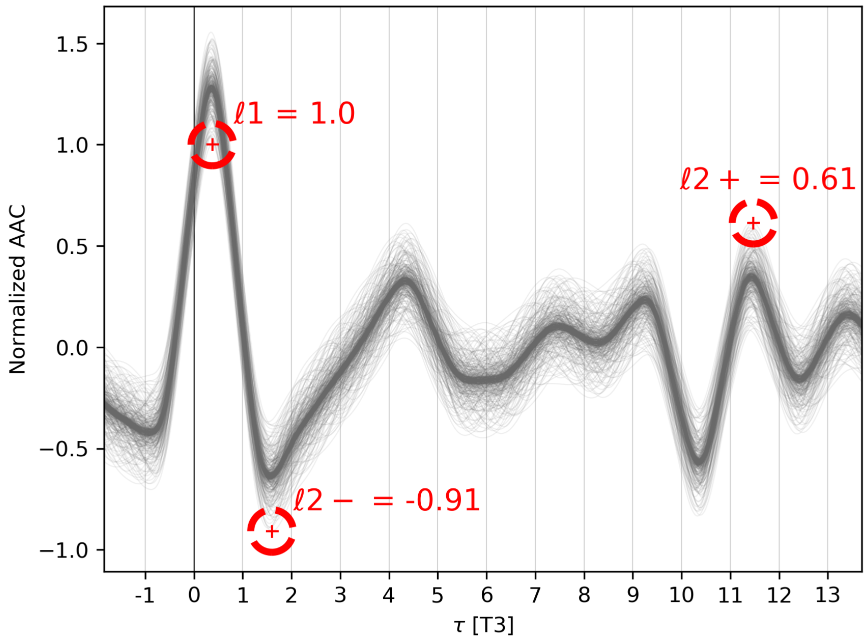


Figure 3.5: Notable points of the Aperiodic Autocorrelation (AAC) of JJJK

(42) The Sample Rate (SR) used is 1 Gigasample(s) per Second (Gs/s) for all our simulations, unless specifically noted otherwise.
(43) IEEE Std 802.3cg-2019 [9] “Table 147–2—DME timings”.

An example output of this process for `JJJK` is shown in Figure 3.5, indicating the location and value of the 3 notable points as well.

Now, let us define the **goodness function** $\mathrm{G}_{\mathrm{AAC}}$ as per Equation (3.6). During the evaluation process, the preamble candidate with a higher value for $\mathrm{G}_{\mathrm{AAC}}$ is considered to be better, as that expresses "flatness" of the curve relative to the main lobe's height, thus it allows better **recognizability** of the pattern, to make it fulfill all 3 of its roles listed and explained in Section 3.1.2.

$$\mathrm{G}_{\mathrm{AAC}} = \min(\mathrm{abs}(\frac{\ell 1}{\ell 2+}), \mathrm{abs}(\frac{\ell 1}{\ell 2-})) \tag{3.6}$$

To ease visual comparison in subsequent parts of the dissertation, the AAC of two preambles will occasionally be plotted on top of each other, in a way that provides the best visibility of relative details. In those cases $\ell 1$ (a.k.a. the normalization value) will be calculated and applied separately for each of the displayed waveforms, this way establishing a visual basis for comparison.

3.3.3.B Goodness of distinguishability

The notion of goodness has been introduced in Section 3.2, while Section 3.3.3 explained simulation that allows weighing **distinguishability**. This property of a waveform comes into play to achieve the following two things, and in the evaluation process it is used typically later on some preamble candidates that have already gone through the recognizability-based evaluation:

- In case the segment may concurrently use more than one preamble at the same time to differentiate between frames with different encoding – for example a frame with and another without Forward Error Correction (FEC) – these two or more preambles shall also be well-distinguishable (between each other);
- To keep separation from the signal form of the BEACON of PLCA [44]: When a PHY is between frames, one of the following two events may occur under normal conditions:
  - Either a new BEACON may be received;
  - Or a new Ethernet frame may be received, which starts with `JJHH` or the new preamble.

As per this, an additional necessary means to judge the fitness of a new preamble candidate is to verify the likelihood of it being mistakenly taken for a BEACON or for `JJHH`. This evaluation uses ACC to understand this property of a waveform.

To achieve this, the ideal form of the preamble being evaluated for distinguishability is fed into Equation (3.3), along with the ideal form of other preambles and/or that of BEACON, and the output of Equation (3.7) – effectively the definite integral of the absolute values of ACC – is used for comparison. The waveform with a lower value for $\mathrm{G}_{\mathrm{AAC}}$ is considered to be better, as it is more distinguishable than the waveform with a higher value for this.

$$\mathrm{G}_{\mathrm{ACC}} = \int_{1-L}^{L-1} |\mathrm{ACC}_{s_a,s_b}(\tau)|\, d\tau \tag{3.7}$$

[44] The final, accepted, form of the BEACON for 10BASE-T1S is `NNNNN`, but at the time of this research it had been `NNNN`, thus we use the latter herein.

3.4 Present research on the preamble

3.4.1 Reproducing earlier results of consensus

The series of previous research results [62, 63, 64, 65] that established basis for evaluation and comparison – using the methods explained in Section 3.3.3.A – have published the results shown in Table 3.1 with respect to the lobe values of `JJHH` and Ga32.

Table 3.1: Notable points of `JJHH` and Ga32 according to previous research

	Ga32	`JJHH`
$\ell 1$	1	1
$\ell 2+$	0.59	0.29
$\ell 2-$	-0.81	-0.67

Previous research never compared the performance of `JJHH` to that of GCSP directly. While the established evaluation method would be capable of producing such a comparison, that would not be a fair evaluation, as GCSP is a considerably longer sequence. Additionally, even if GCSP were much better than competing patterns, it has the considerable disadvantages later explained in Section 3.4.3.

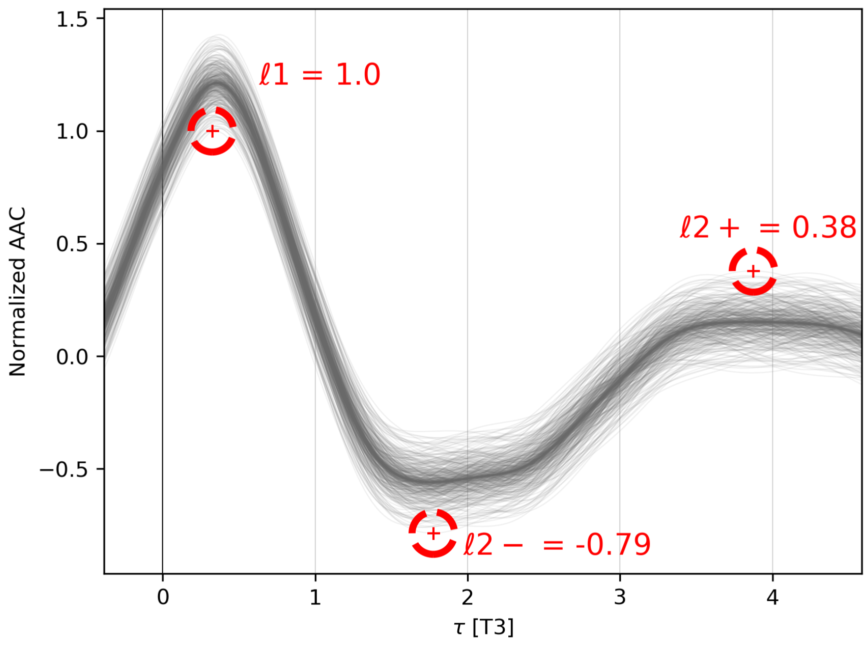


Figure 3.6: Notable points of the Aperiodic Autocorrelation (AAC) of `JJHH`

Research presented here first aimed at reproducing the figures shown in Table 3.1 with acceptable precision. The first attempt to achieve this was through the direct application of the evaluation environment explained in Section 3.2 over `JJHH`, the outcome of which is displayed in Figure 3.6. As visible, the differences between the values for $\ell2+$ and $\ell2-$ listed in Table 3.1 and those shown in Figure 3.6 are quite considerable. While a small difference may be explained by the obvious mismatch between the actual values of AWGN being used in the two sets of simulations that generate s_3 from s_2, it has been verified that what is observable here is beyond that magnitude[45].

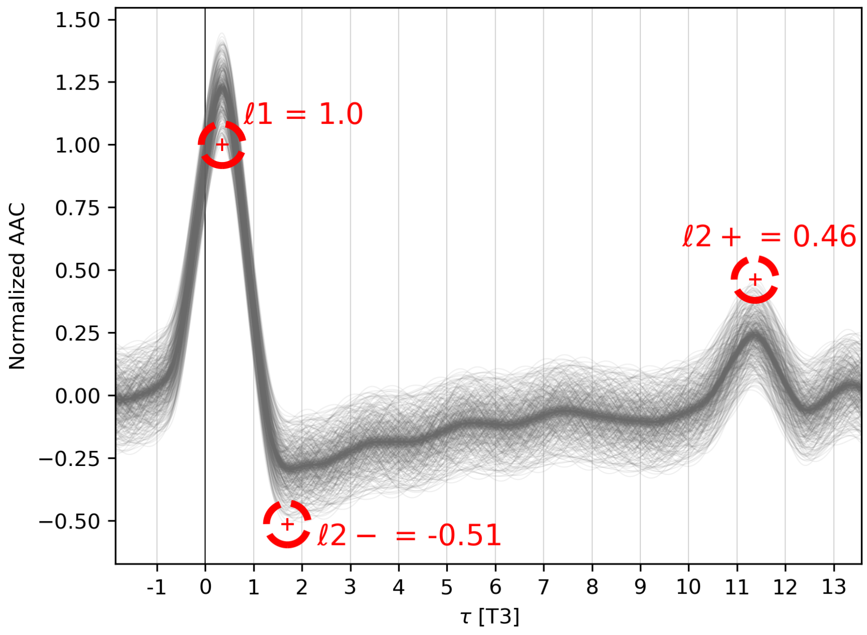


Figure 3.7: Notable points of the Aperiodic Autocorrelation (AAC) of Ga32

The next step has been the execution of the evaluation environment over Ga32, which also produced surprising results, as shown in Figure 3.7. The literal values produced by new simulations are summarized in Table 3.2, while Figure 3.8 and Figure 3.9 visualize all the differences in a comparative manner. The numerical differences are shown in Table 3.3. What is notable here beyond differences between earlier and new results for `JJHH` and Ga32 is the fact, that if the new values were to be found correct, it would result in a definite change in the goodness order between these preambles, making Ga32 to be superior over `JJHH`.

3.4.2 Understanding the reasons for the differences

To try to understand the reasons for the differences, first a thorough verification of the simulation environment and its implementation has been done. As this provided no explanation, a contact has been taken the authors of the results in question [65].

[45] Nevertheless, to be able to produce consistent and reproducible results within the research presented in the dissertation, the simulation environment has been implemented in a way, that always uses the same seed value for the underlying Random Number Generator (RNG), thus all new simulations do rely on the same, and reproducible, numeric values for AWGN, therefore new results are expected to be consistent among themselves.

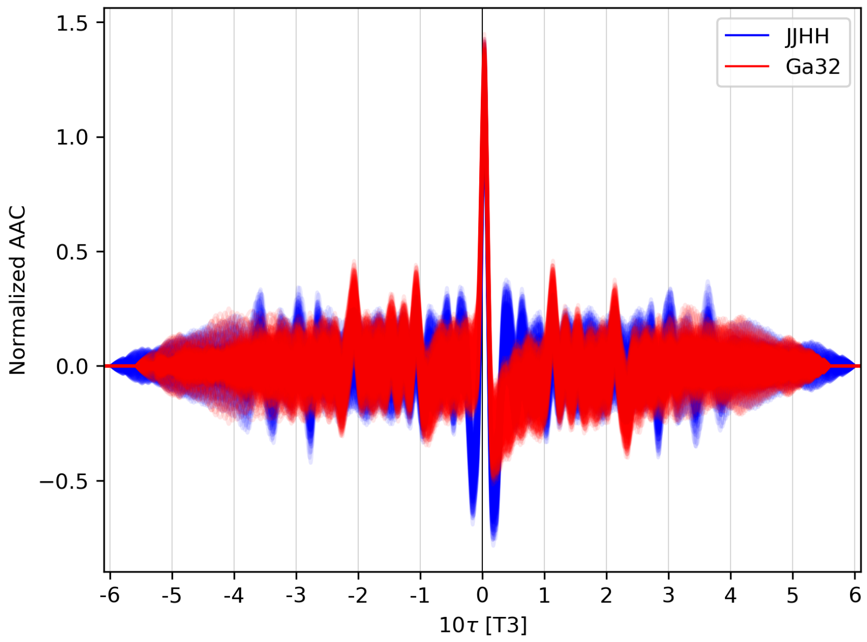


Figure 3.8: Comparison of Aperiodic Autocorrelation (AAC) of `JJHH` and Ga32

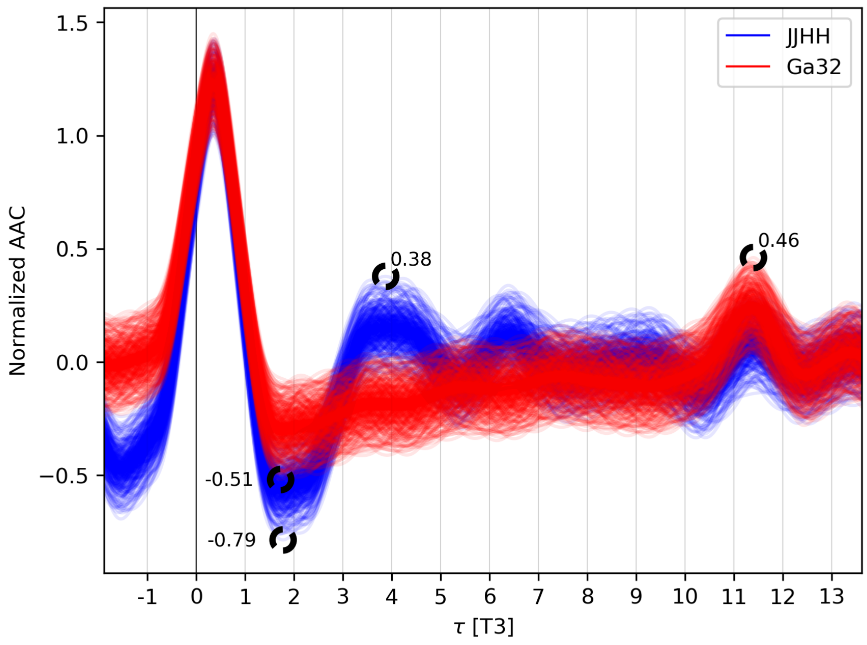


Figure 3.9: Notable points of the Aperiodic Autocorrelation (AAC) of `JJHH` and Ga32

Table 3.2: Notable points of JJHH and Ga32 according to new simulations

	Ga32	JJHH
$\ell 1$	1	1
$\ell 2+$	0.46	0.38
$\ell 2-$	-0.51	-0.79

Table 3.3: Comparison of earlier [65] and new results on the goodness of recognizability of some preambles

	$\mathrm{G_{AAC}}$	
	Previous results	New results
Ga32	1.23	1.96
JJHH	1.49	1.27

3.4.2.A Background of differences for JJHH

Tight cooperation and careful comparison of the two simulation environments revealed that the differences were due to the following two factors:

- Earlier research used the considerably lower sampling rate of 100 Ms/s instead of the 1 Gs/s used here. The lower resolution of the bilinear transform when the Nyquist Frequency is near the corner frequency [46] is responsible for a considerable part of the observed differences;
- For an unknown reason, earlier research did not apply step #4 of the channel model (namely the MCM), in other words in those simulations $s_3 = s_2$, which made the system ignore the influence of the actual channel, which also contributed to the observed differences.

To confirm these findings, a new simulation has been set up to follow the method used by fellow researchers. The output of that effort produced the results shown in Figure 3.10. As visible, within the error imposed by the rounding to two decimal places, the new results precisely match those published earlier [65] for JJHH, confirming that – while the modified environment may not necessarily give better basis for comparison – it does produce matching results for that part of the research that have earlier been published.

3.4.2.B Background of differences for Ga32

While at this stage, the differences in findings for JJHH were well understood, this was clearly not the case for Ga32: it is visible from Figure 3.9 and Figure 3.10 that none of our results for Ga32 match those published earlier, therefore it was still not decidable which of these preambles were indeed better with respect to recognizability.

To understand this independent set of differences, an additional round of joint comparison of the two simulation environments revealed that earlier work [65] has not

[46] For the LPF this is expressed by $\frac{3\cdot 10^7}{10^8/2} = 0.6$ for SR = 100 Ms/s, and by $\frac{3\cdot 10^7}{10^9/2} = 0.06$ for SR = 1 Gs/s.

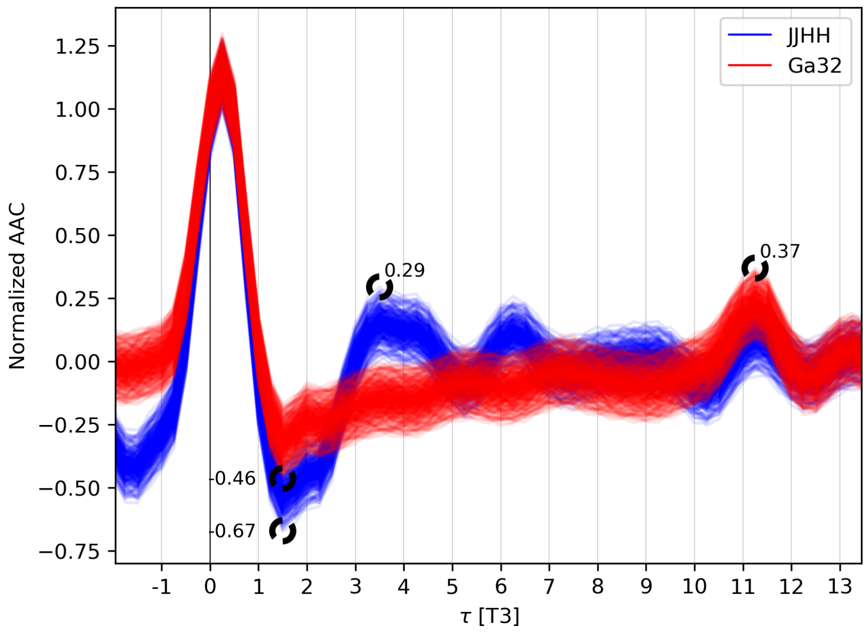


Figure 3.10: Notable points of the Aperiodic Autocorrelation (AAC) of `JJHH` and Ga32 with SR = 100 Ms/s

formally calculated AAC for Ga32, but used instead an assumed version of it from [64]. This process involved manual scaling of images, which led to the incorrect results with respect to AAC for Ga32, which then subsequently led to the incorrect conclusions be drawn with respect to the relative order of these two preamble candidates. It is interesting to point out – especially in retrospect – that it is visible in the earlier presentation that the overall shape of AAC for Ga32 was approximately correct, however its magnitude was off.

3.4.2.C Conclusions on the differences

Due to the findings explained in the previous sections, fellow researchers later revised their work and confirmed the following:

- Indeed the new simulation method and results were correct. This confirmation is important as additional new results presented subsequently in Section 3.4 were to be based on this established method;
- Several intentional and unintentional deviations from the agreed channel model were made during previous research, leading to incorrect numerical values, as well as an incorrect conclusion;
- Indeed – purely from the perspective of recognizability – Ga32 **is better than both `JJJK` and `JJHH`**.

3.4.3 Looking for the best DME preamble

While at this point it is clear that Ga32, Gb32, and GCSP do outperform all DME-based candidates with respect to recognizability, it is worth pointing out that the earlier have notable disadvantages as well:

- While Ga32 and Gb32 are slightly shorter than DME-based candidates ($1.28\mu s$ versus $1.6\mu s$, respectively), GCSP is considerably longer than these ($4.48\mu s$), therefore not only direct comparison would be unfair, but a longer constant part in the preamble would leave very limited number of bits to be able to relay for example seed or other synchronization information for the scrambler. This has also been recognized by the researchers proposing Ga32 and Gb32, but it has been argued [64] that remaining bits would still be sufficient to implement a scrambler to achieve acceptable Electromagnetic Interference (EMI) performance;
- Unlike `JJJK` (and all other 5B symbols proposed to be used by Clause 147), Ga32, Gb32, and GCSP are not DME-based binary signals that consist of only a negative and a positive values, but these are ternary sequences (40) that are composed of a mixture of negative, zero and positive signal values. Two of the key features of DME modulation are that clock recovery and detection of end of transmission (via code-violation) are trivial, allowing for simpler receiver architecture even in presence of considerable noise and jitter. In contrast with this Ga32, Gb32, and GCSP would have required its own set of high-speed circuitry, considerably increasing complexity of an implementation [64, 65].

While analyzing the performance of Ga32, Gb32, and GCSP have been essential in establishing and validating the preamble candidate evaluation methods, for the above-mentioned reasons **non-DME sequences** have been excluded from the search space for further research.

As later shown in Table 4.2, the standard defines only $16 + 8 = 24$ out of the total possible of $2^5 = 32$ 5B symbol values, and the 5B symbol value of `I` has been assigned special – inter-layer – signaling functions by the standard (47). For these reasons, this theme of our research aimed at being concluded through the following steps:

1. The following 7 DME-encodable 5B symbols with no 4B mapping defined by the standard were to be used for further search: `J`, `K`, `T`, `R`, `H`, `N`, and `S`. During the process, all $7^4 - 1 = 2400$ (48) possible 4-long 5B sequences were to be evaluated through the simulation environment previously established and validated;
2. The output of this search would be sorted according to the $\mathrm{G_{AAC}}$ values each sequence produces;
3. If any sequence better than `JJHH` were to be found, those would be further analyzed for their properties of distinguishability, according to the method described in Section 3.3.3.B.

(47) IEEE Std 802.3cg-2019 [9] "147.3.2.1 PCS Transmit overview".
(48) As described in Section 3.3.1, the BEACON signal is composed of only `N` symbols.

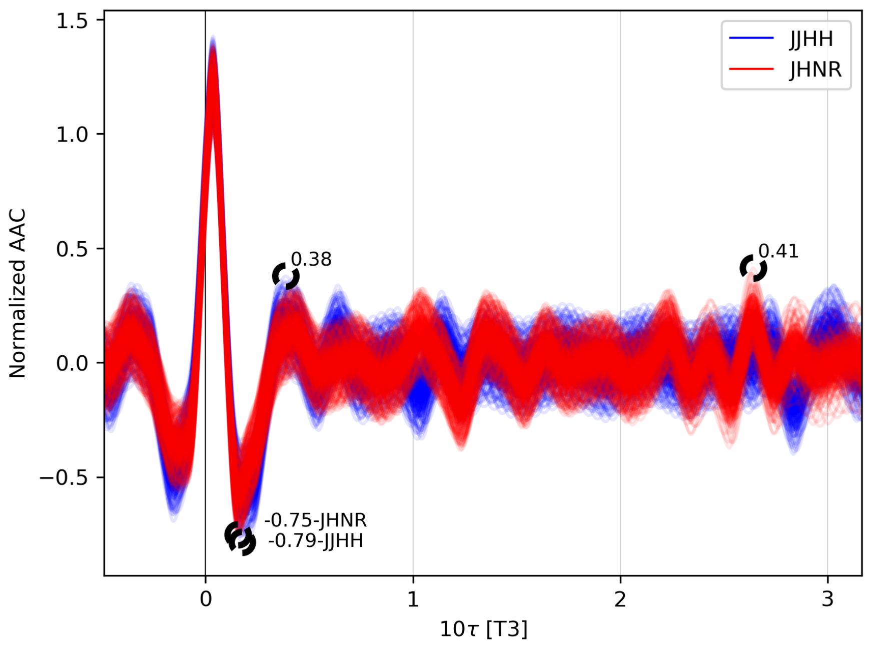


Figure 3.11: Notable points of the Aperiodic Autocorrelation (AAC) of `JHNR` and `JJHH`

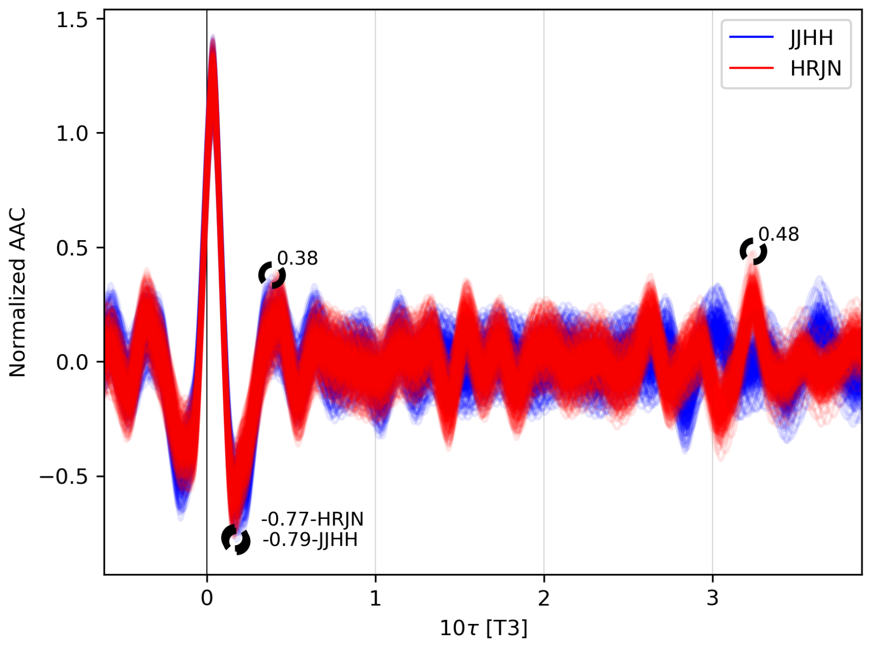


Figure 3.12: Notable points of the Aperiodic Autocorrelation (AAC) of `HRJN` and `JJHH`

3.4.3.A Preambles better than JJHH

The method described in Section 3.4.3 has led to the interesting conclusion, that while JJHH is indeed one of the best DME-based preambles that use 5B symbols defined by the standard [49], JHNR [50] and HRJN [51] are the two best patterns with respect to recognizability, the details of which are shown in Figure 3.11 and Figure 3.12 (respectively).

As visible from the two G_{AAC} values, these new constant patterns are close to each other with respect to recognizability, however there is an additional consideration that helps making the distinction. This metric is the PCS_RX complexity, which in case of HRJN is expected to be smaller when HRJN is used in conjunction with JJHH, as this pattern has different 5B symbol at each of the 4 symbol positions, therefore PCS_RX that does not use correlators to detect preambles can make the distinction quicker with HRJN.

3.4.3.B Distinguishability of the new preamble candidates

The final part of the research on the preambles will follow the reasons and methods described in Section 3.3.3.B, the purpose of which is to understand whether the introduction of the newly proposed preambles would create problems with distinguishing between those, JJHH, and NNNN [41].

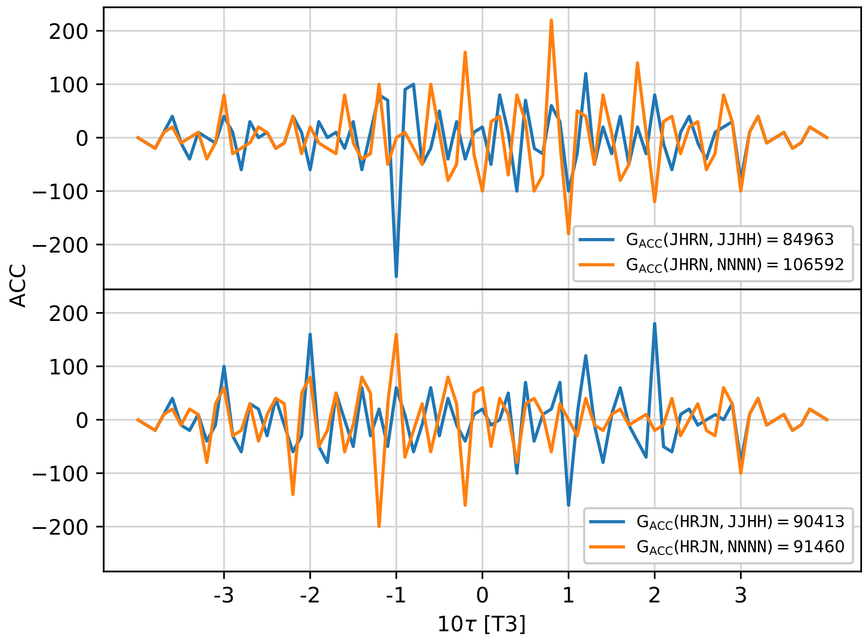


Figure 3.13: Aperiodic Cross Correlation (ACC) of the best preamble candidates and constant patterns already in the standard

[49] JJHH came out to be the 3rd best.
[50] G_{AAC}(JHNR) = 1.33.
[51] G_{AAC}(HRJN) = 1.30.

As apparent from Figure 3.13, with respect to ACCs, `HRJN` comes out first due to the following two reasons:

- The average of the two G_{ACC} values for `HRJN` is smaller than that for `JHNR`;
- Also the absolute values of the peaks in the ACC curves for `HRJN` are smaller than those for `JHNR`.

3.5 Conclusions on the 10BASE-T1S preamble

This theme of research can be summarized as follows:

- The simulations [65] based on which the goodness of Ga32 and `JJHH` have been evaluated and `JJHH` has been selected as the preamble for 10BASE-T1S included errors, leading to incorrect observations and conclusions.

  What became apparent, is that within the evaluation method of consensus, Ga32, Gb32, and GCSP do indeed outperform `JJHH`, particularly on a channel with maximum IL and maximum total capacitance for 10BASE-T1S.

  As per the explanation given in Section 3.4.3.B however, non-DME patterns are excluded from further evaluation;
- All the examined DME preamble pattern constants formed the following list in decreasing order of goodness of recognizability: `JHNR`, `HRJN`, immediately followed by `JJHH`, with `JHNR` and `HRJN` also being close to each other;
- With respect to distinguishability from `JJHH` and the BEACON, and support for low-complexity PCS_RX implementations `HRJN` does outperform `JHNR`;
- To summarize all aspects listed above, `HRJN` came out to be the best alternative sync pattern, capable of achieving any of the following goals:
  - Used alone, to implement any new low-complexity PHY that operates on top of DME modulation;
  - Extend the capabilities of 10BASE-T1S by enabling features like Operations, Administration, and Maintenance (OAM), Out of Band (OoB) signaling, etc., in parallel with normal exchange of Ethernet frames, without causing interference.

The statements above serve as a conclusion for this theme of the research.

Chapter 4 Backward-compatible Forward Error Correction of burst errors and erasures for 10BASE-T1S

4.1 Introduction

Research presented in this chapter first focuses on the reason why there is no error correcting capability in 10BASE-T1S, then it shows that to meet certain objectives of IEEE Project 802.3da [29] (P802.3da) it may be unavoidable to implement an Forward Error Correction (FEC) into the successor of 10BASE-T1S, named 10BASE-T1M.

Subsequently, the chapter walks the reader through the FEC solution candidates and shows why it is optimal to build on a systematic Reed-Solomon Error and Erasure Correcting Code (RSC). The major, part of the chapter points out why no well-known FEC scheme can be used directly, and proposes a set of novel techniques, through the application of which 10BASE-T1M may be equipped with a backward-compatible FEC that allows it to meet all related objectives of P802.3da.

In the latter part of the chapter, the construction and operation of an example implementation of the proposed theoretical scheme is shown, which is followed by a comparison of the novel $\{c, u\}$ coding scheme to other solutions, while the chapter is concluded by the evaluation of the performance of this scheme.

4.1.1 Why FEC?

As explained in Section 2.4, there are a number of project objectives that demand noise – particularly impulse noise – resilience in 10BASE-T1M. Moreover, in P802.3da majority of participants are from the industrial, automotive, and automation fields of the engineering science, well accustomed to networks dominated by impulse noise.

In these environments, one of the most common challenges is dealing with the trade-off between performance, complexity, and costs. As an example, while some of the problems related to stationary and impulse noise sources that couple to the media, may be tackled by applying shielding to the cable, this not only raises costs and complexity, but also pose yet unsolved technological challenges in case of a mixing segment, including the difficulties related to maintaining the continuity and integrity of the shielding. Further, it has been discussed that even under ideal conditions, impulse noise sources common to industrial environment tend to carry energy that is sufficiently large to cause artifacts in the single-ended signal domain.

Due to these concerns, one of the most natural ways of dealing with impulse noise is a **burst error correcting FEC**. As the signal swing of 1 V (0.5 V_{RMS}) specified by the standard (29) is considerably lower than what silicon technologies applicable for 10BASE-T1S may tolerate on the receive path, the additional headroom (margin) optionally allows detection of the presence of noise already in the analog domain, this way enabling the possibility of optionally using **erasure** detection as well through side-channel information.

While there are other means to counter the effect of the noise beyond using an FEC, those were not applicable in this case, as explained in Section 2.5.2. In short, FEC with optional erasure detection is a very natural direction of evolution for 10BASE-T1S.

4.1.2 The roots of problem being solved

Implementing a general-purpose FEC for 10BASE-T1S would not be particularly difficult or interesting from academic perspective, as adding an FEC that is capable of dealing with impulse noise to a frame-based networking solution has ample history. The difficulty of the problem at hand is rooted in objective #4 of P802.3da [27] explained in Section 2.5.2, which requires the solution to remain interoperable with 10BASE-T1S. The technical details of, and finer-grained requirements creased by, the interoperability will be explained in Section 4.3, however we would like to give here the outline of the difficulties it poses.

Discussions whether objective #4 is indeed worth keeping are ongoing in P802.3da, generating output even at the time of writing of the dissertation [48, 49, 50]. As the fate of objective #4 is yet to be decided, results presented in this section will work under the assumption that interoperability with Clause 147 is a must.

To remain interoperable with 10BASE-T1S while maintaining low complexity and a throughput that asymptotically converges to the the nominal 10 Megabit(s) per Second (Mb/s) require the payload of a FEC-enhanced frame to look almost like that of a regular 10BASE-T1S frame, such that each 4B user data symbol conveyed by the Media Access Control (MAC) be encoded into not more than a single 5B channel symbol at average. Moreover, enabling the optional use of interleaving to achieve the required per-frame configurable error resilience puts additional constraints on the FEC as later explained in Section 4.5.2. These requirements together force the implementation to use nothing more than the very small remaining (unused) redundancy present in the 4B/5B mapping defined by Clause 147, displayed in Table 4.2. It will later be shown that the amount of this unused information is ≈ 0.8 bits per 5B symbol.

These requirements alone however are still not difficult to meet, as real-life burst Error Correcting Codes (ECCs) do exist [66, 67] that operate on comparable or even more restrictive rates. What complicates the problem further is that this limited amount of information is not fully available for the ECCs, as these very same information space is also used to avoid certain 5B symbols from appearing anywhere in the codewords. The details of this will later be explained in Section 4.3.4, however it can be said already here, that any straightforward – for example bit stuffing-based – approach would require 1 bit per 5B symbol, immediately exceeding the limit explained above. This property of the problem drastically limits the applicability of any known low-complexity coding techniques, as later demonstrated in Section 4.7.

The best of our knowledge, no coding technique exists that would be directly applicable to this problem. Moreover, we also believe that the scheme outlined in the dissertation may be suitable to solve other (similar) problems, and it may also be extended to further improve its performance. A glimpse to these future opportunities is given in Section 5.2.

4.1.3 Connection with the research on preamble

Chapter 3 has introduced a method, that allowed new preambles to be used alone, or in conjunction with the current preamble of 10BASE-T1S defined by Clause 147 of IEEE Std 802.3cg-2019 [9] to implement additional features providing error resilience or functions related to that [52].

After the conclusion of that research theme however, a new project – namely P802.3da – has been approved with a new set of objectives [31]. One of the most notable among these is objective #4 [27]. This objective not only required the new Physical Layer Device (PHY) – to be referred to as 10BASE-T1M – to preserve the

[52] Earlier attempts to propose features [68] that would have allowed inter-PHY signaling, multi-party Auto-Negotiation (AN), Operations, Administration, and Maintenance (OAM), etc. failed, despite the clear need for these functions.

preamble JJHH of 10BASE-T1S, but it also prevented parallel preambles to be defined. This is because if an implementation chooses to implement Physical Coding Sublayer's Receive Function (PCS_RX) literally [53], then it will be prone to incorrect detection of the Beginning of Frame (BoF).

4.1.4 Research and verification methods

All observations made in this chapter are based on the standard text, as well as a verbatim implementation of the Finite State Machines (FSMs) defined in Clause 147 and Clause 148 [46] done for the purpose of the research. Additionally, the proposed FEC scheme has been fully implemented without using any external libraries [47]. More on these aspects of the work can be found in Section B.2.

4.2 Channel model

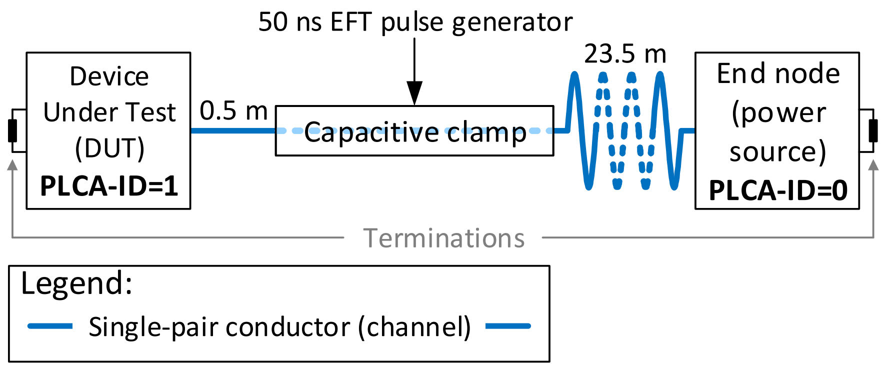

50 ns EFT pulse generator
23.5 m
Device Under Test (DUT) PLCA-ID=1
0.5 m
Capacitive clamp
End node (power source) PLCA-ID=0
Terminations
Legend:
Single-pair conductor (channel)

Figure 4.1: IEC 61000–4–4 Electrical Fast Transient (EFT) test setup

According to the explanation in previous sections, the research presented in this chapter assumes a **binary burst error channel with optional erasure detection**.

One of the most commonly accepted setups for this type of channel in the industry to test a system's response to burst stimuli is the IEC 61000–4–4 Electrical Fast Transient (EFT) test [27], the application of which has been requested in P802.3da [69]. The outline of the test setup is shown in Figure 2.2. In there, the medium (in other words the "channel") is exposed to a high-energy source of transient impulse noise, that – due to the coupling between the differential and common mode signals – manifests itself as **burst** disturbance from the perspective of the receivers.

The test setup shown in Figure 2.2 – which is analogous to the schematic arrangement of Shannon's general communication system [70] – exposes the channel to a sequence of 50 ns stimuli. Given the 80 ns Differential Manchester Encoding (DME) bit time of 10BASE-T1S, one may expect that no more than two DME bits may be disturbed by a single 80 ns pulse, it has been shown [71] that in practical systems however, due to the common mode impedance, the EFT pulse coupled to the differential pair may cause errors on the channel which are up to 450 ns in duration, or six DME bit times, per stimulus pulse.

[53] IEEE Std 802.3cg-2019 [9] "Figure 147–7—PCS Receive state diagram, part a" and "Figure 147–8—PCS Receive state diagram, part b".

While this is a value a specific FEC scheme may be built around, such approach would considerably limit the broader use and value of the solution. For this reason, the aim of the research has been to propose a general solution, where the scheme allows per-frame configurable [72] burst error resilience.

4.3 Backward-compatibility

While a short introduction to the background of backward-compatibility has already been given in Section 4.1.2, in this section a complete definition is provided.

4.3.1 The reasons for requiring backward-compatibility

Backward-compatibility – or in the wider context “interoperability” – is a key concept in the Ethernet world. Making PHYs of different specification able to interact as long as those are inter-connectable via the same medium is typically based on a feature called Auto-Negotiation (AN) [3] and it is integral part of the standard (54). Using AN backward-compatibility is achieved through a two-actor negotiation process, through which the two ends of a Point-to-Point (Pt2Pt) link agree on the highest level of functionality common to the two PHYs. For example, a 100 Mb/s and a 1 Gigabit(s) per Second (Gb/s) PHY is expected to be able to establish and maintain a 100 Mb/s Pt2Pt through AN.
The reason why this is not currently applicable in case of 10BASE-T1M is two-fold:

- AN works only over Pt2Pt links, and no specification for AN over Multidrop (MDROP) exists;
- Even if AN were capable of working over MDROP, not only it would be required to be capable of dealing with challenges like node joins and leaves (including churn), it would also limit all participants to the least common denominator of features, preventing PHYs with FEC capability to establish noise-resilient communication between themselves if there was even one non-capable device connected to the mixing segment. This limitation would make such a system unable to deal with noise local to some of the receivers, which is a common use case in industrial environments.

4.3.2 What backward-compatibility entails?

4.3.2.A General definition

In short, backward-compatibility requires that the FEC-enabled frames are formulated such a way that Clause 147- and Clause 148-compatible (10BASE-T1S) nodes are able to coexist on the same shared medium as currently specified in IEEE Std 802.3cg-2019 [9]. Obviously, 10BASE-T1S would not be aware of the presence of the FEC, neither should it be allowed to attempt to receive such an enhanced frame. Guaranteeing the latter is also part of implementing backward-compatibility.

(54) IEEE Std 802.3-2018 [2] “28. Physical Layer link signaling for Auto-Negotiation on twisted pair”.

4.3.2.B Requirements in technical terms

A 10BASE-T1M would need to meet the following criteria to remain backward-compatible with 10BASE-T1S the way explained in the previous section:

- A 10BASE-T1S PHY has to be able to receive the bit stream predictably, therefore the Physical Medium Attachment Sublayer (PMA) of 10BASE-T1M must transmit DME-encoded bits at a rate of 12.5 Mb/s, resulting in a 5B symbol rate of 2.5 MHz, using analog signals that are defined for 10BASE-T1S [(55)];
- 10BASE-T1S's PCS_RX has to manifest predictable behavior when receiving a frame from 10BASE-T1M. Part of this is guaranteeing that 10BASE-T1S would not attempt to decode the frame, which could possibly lead to false positive acceptance of the encapsulated, transcoded data;
- Transmission of FEC-enabled frames should not disrupt the Physical Layer Collision Avoidance (PLCA) cycle of the mixing segment and any of its stations;
- The data rate of 10 Mb/s must be maintained at the Media Access Control/Physical Signaling (MAC/PLS) interface, preferably without the need for additional buffers within the PHY. This is desirable, because the size of such buffers would need to scale with the size of the largest Ethernet frame the PHY is capable of transmitting.

In short, the consequence of these is that a FEC-enabled frame needs to be a DME-encoded with a rate of no less than $4/5 = 0.8$ using the PMA transmitter electrical specification of Clause 147, thus the solution must be a **rate-preserving** ECC.

4.3.2.C Additional preferences

An additional, implicit, requirement is to avoid diverging from the design criteria that brought 10BASE-T1S to life: the coding scheme is expected to be implementable with low complexity and its power consumption should not considerably increase the power budget requirements of the system.

4.3.3 Details of backward-compatibility per sublayer

The technical requirements listed in Section 4.3.2.B already constrain the PMA of 10BASE-T1M, therefore in this section we focus on the backward-compatibility with the Physical Coding Sublayer (PCS) [(7)] and PLCA [(8)].

4.3.3.A Backward-compatibility at the PCS

Detailed analysis of the FSMs of 10BASE-T1S's PCS_RX [(43)] and simulations showed that to maintain predictable behavior, the 10BASE-T1S has to be locked into a well-defined cycle (loop) of state(s) for the duration of the reception of the frame. Further inspection of the relevant clauses of the standard led to the simplified summary of the cycles present in the PCS_RX of 10BASE-T1S shown on Figure 4.2. The two existing cycles are as follows:

- The cycle highlighted in blue that is comprised of the sequence of states of `WAIT_SYNC`(→`SYNCING`(→`COMMIT`(→`WAIT_SSD`)))→`WAIT_SYNC`→..
  The role of this sequence is the formal detection of the preamble `JJHH`, as discussed in Section 3.4.3.A;

[(55)] IEEE Std 802.3cg-2019 [9] "147.5.4 Transmitter electrical specification".

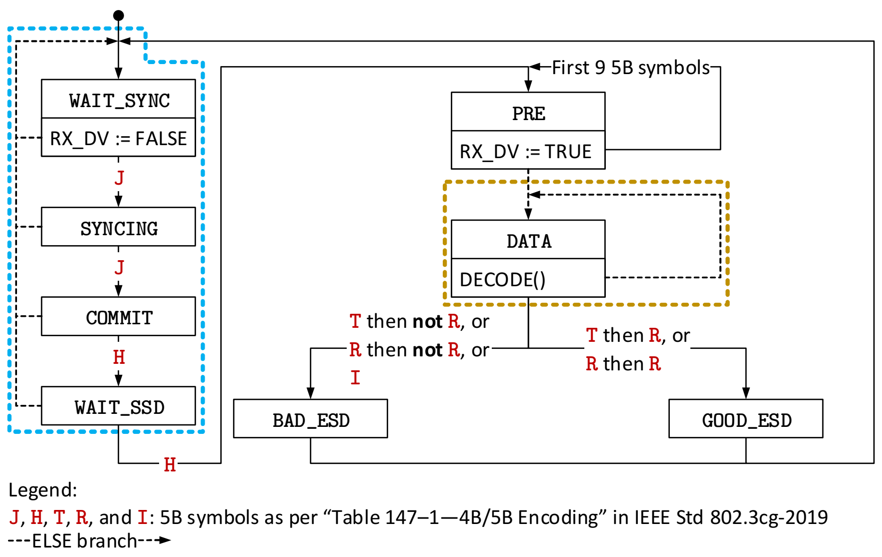


Figure 4.2: Summary of both cycles in the PCS_RX of 10BASE-T1S

- The cycle denoted in brown is a single state loop within `DATA`, the role of which is to receive the variable-size payload of a frame, that is started by a successful preamble detection and ends in receiving an End Sequence Delimiter (ESD).

Further analysis work revealed that cycle #1 is not useful for locking PCS_RX, as it interferes with the operation of PLCA by deasserting RX_DV, based on which PLCA is counting its cycles, the details of which can be found in Section 4.3.3.B.
Accordingly, only cycle #2 is useful for maintaining control of PCS_RX, the conditions of which are summarized in Table 4.1.

Table 4.1: Exit scenarios for state `DATA` of PCS_RX of 10BASE-T1S

	$RX_{n-3} = \mathtt{I}$	$RX_{n-3} = \mathtt{T}$	$RX_{n-3} = \mathtt{R}$	ELSE	
				$RX_{n-1} = \mathtt{R}$	$RX_{n-1} \neq \mathtt{R}$
$RX_{n-2} = \mathtt{T}$	*BAD*	–	–	–	*BAD*
$RX_{n-2} = \mathtt{R}$	*BAD*	*GOOD*	*GOOD*	–	*BAD*
ELSE	*BAD*	–	–	–	–

Legend:
GOOD: The transition `DATA`→`GOOD_ESD`
BAD: The transition `DATA`→`BAD_ESD`
–: No transition from state `DATA`

Additionally, to guarantee that a non-FEC-enabled PHYs discard the frame without attempting to decode it, the state `DATA` must be exited through the state `BAD_ESD`.

4.3.3.B Backward-compatibility with PLCA

Analysis of the FSMs of PLCA Control Function (PLCA_CTRL) [(56)], those of PLCA Data Function (PLCA_DATA) [(57)], and simulations of these showed that compatibility at PLCA can be achieved if during the reception of an FEC-based frame, PLCA_CTRL of the 10BASE-T1S is locked in `RECEIVE` state.

4.3.3.C Summary of backward-compatibility per sublayer

To sum these requirements, it can be stated that to achieve backward-compatibility at both PCS and PLCA of 10BASE-T1S, the following are the necessary and sufficient conditions:

- The PCS_RX FSM of the receiver is locked in `DATA` state, as explained in Section 4.3.3.A and in Section 4.3.3.B;
- The state `DATA` is left through the state `BAD_ESD`, this way guaranteeing the discarding of the frame in non-FEC-enabled PHYs.

This works, because if these conditions are met, from the perspective of a 10BASE-T1S PHY an FEC-enabled frame looks very much like a regular 10BASE-T1S frame, with explicit indication of transmitter-side error.

4.3.4 Forbidden Symbols

From Section 4.3.3 it is visible that to meet the conditions defined in Table 4.1, an FEC-enabled frame has to avoid the appearance of three 5B symbols – namely `T`, `R`, and `I` – to remain backward-compatible with 10BASE-T1S.

As these three 5B symbols must never be produced by FEC encoder while PCS_RX is expected to stay in `DATA` state, we are going to refer to these as **base Forbidden Symbols (FSs)**, and the remaining 5B symbols that are not forbidden are going to be called **base Admissible Symbols**.

4.3.5 Summary of all backward-compatibility requirements

To summarize the criteria for an FEC-enabled frame to be backward-compatible with 10BASE-T1S with PLCA, it has to be constructed the following way:

- It has to be a DME-encoded stream of 5B symbols, with a rate of no less than $4/5 = 0.8$, according the PMA transmitter electrical specification in Clause 147;
- It must use the unused redundancy present in the 4B/5B mapping defined by the standard, as shown in Table 4.2, and – in the 5B domain – it must avoid the appearance of FSs on the output of the FEC encoder.

  Let operator $|\mathcal{C}|$ denote the cardinality of set $\mathcal{C}$ and $\mathcal{F}_\mathrm{b}$ represent the set of base FSs. Now, as $|\mathcal{F}_\mathrm{b}| = 3$, the information quantity carried by the unused redundancy is $\log_2(32 - 3) - 4 \approx 0.86$ bits per 5B symbol.

(56) IEEE Std 802.3cg-2019 [9] “Figure 148–3—PLCA Control state diagram, part a” and “Figure 148–4—PLCA Control state diagram, part b”.
(57) IEEE Std 802.3cg-2019 [9] “Figure 148–5—PLCA Data state diagram, part a” and “Figure 148–6—PLCA Data state diagram, part b”.

It is important to point out that by taking the unused redundancy present in the 4B/5B mapping in use, the new code will also eliminate the effect of certain properties the current mapping has with respect to the maximum number of consecutive zero and one bits in the control 5B symbols, and those that have 4B mappings. This however would not degrade the performance of the new code (for example with respect to its synchronization capabilities), as in case of 10BASE-T1S, 4B/5B is established not on top of Non-Return-to-Zero (NRZ) modulation, but on DME, which alone is a rate$-\frac{1}{2}$ Run-length Limited (RLL) code [73], offering better synchronization properties than those possibly eliminated by the the proposed scheme.

Further, the electromagnetic emission properties of the FEC-enhanced frame are also expected to be at least as good as that of the original code, due to the inclusion of more 5B codewords, and the maintained presence of the multiplicative scrambling as per the explanation presented in Section 4.4.1.

4.4 Other features in conjunction with the FEC

4.4.1 Multiplicative scrambling

10BASE-T1S has a 17-bit multiplicative scrambler [74] defined for it (58), the purpose of which is to flatten the emission spectrum of the transmitter, to keep it within the defined limits (59) even in the presence of so-called “killer packet” that repeatedly carry same or similar content.

Keeping the scrambler for FEC-enabled frames is not a must, however – right along the original reasons why scrambler was included in the first place – it is much preferred, as the FEC itself would not be able to eliminate the killer packets, it would just make intentionally “forging” such patterns more cumbersome. As the scrambler of 10BASE-T1S is based on a Linear-Feedback Shift Register (LFSR) with the fixed polynomial of $g(x) = x^{17} + x^{14} + 1$, if scrambler were applied on the output of the FEC encoder in the transmit direction, then it would cause error multiplication, essentially decreasing the performance of the underlying FEC.

For these reasons, the appropriate placement of the scrambler in relation with the FEC components is shown in Figure 4.3.

4.4.2 4B/5B encoding and its timing

Std 802.3cg-2019 [9] “Table 147–1—4B/5B Encoding” defines a specific mapping between the 4-bit nibbles (4B symbols) conveyed at the MAC/PLS and the 5-bit nibbles (5B symbols) used on the media through DME modulation. The complete mapping is shown in Table 4.2, the details of which are as follows:

1. The 16 of the 32 5B symbols listed in the leftmost and central columns in Table 4.2 are reserved to map to all the 16 4B user data symbols;

2. The 8 of the remaining 16 5B symbols listed in the rightmost columns in Table 4.2 are assigned control functions of the PCS and PLCA [38, 75];

3. The remaining 8 5B symbols listed in the notes section in Table 4.2 are left unassigned and unused.

(58) IEEE Std 802.3cg-2019 [9] “147.3.2.8 PCS Transmit/Self-synchronizing scrambler” and “147.3.3.8 PCS Receive/Self-synchronizing scrambler”.

(59) IEEE Std 802.3cg-2019 [9] “147.5.4.4 Transmitter Power Spectral Density (PSD)”.

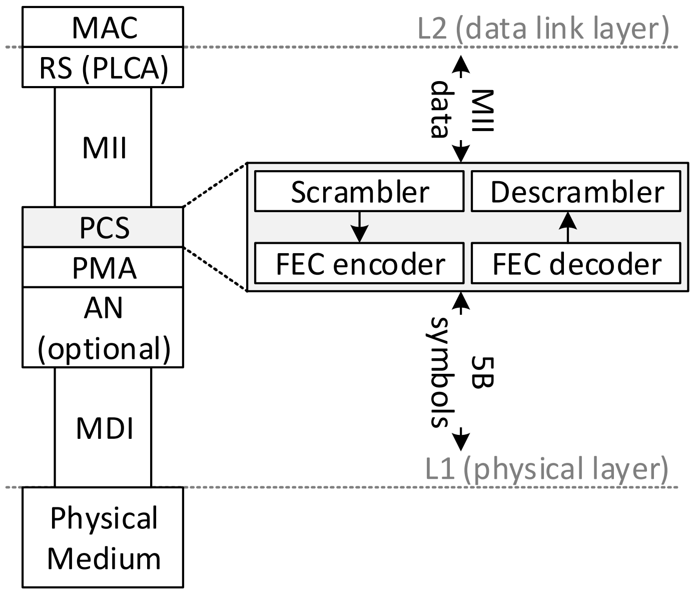


Figure 4.3: Preferred layering of functions in an FEC-enabled PHY, including the proper placement of the scrambler

Table 4.2: The 4B/5B mapping defined by Clause 147

Name	4B	5B	Name	4B	5B	Name	4B	5B	Special function(s)
0	0000	11110	8	1000	10010	I	N/A	**11111**	SILENCE
1	0001	01001	9	1001	10011	J		11000	SYNC and COMMIT
2	0010	10100	A	1010	10110	K		10001	ESDERR
3	0011	10101	B	1011	10111	T		**01101**	ESD and HB
4	0100	01010	C	1100	11010	R		**00111**	ESDOK and ESDBRS
5	0101	01011	D	1101	11011	H		00100	SSD
6	0110	01110	E	1110	11100	N		01000	BEACON
7	0111	01111	F	1111	11101	S		11001	ESDJAB

Notes:
– 4B and 5B values are displayed in binary format;
– The three 5B symbols highlighted in **bold** are of special interest as later shown in Table 4.3;
– The 5B values of 00000, 00001, 00010, 00011, 00101, 00110, 01100, and 10000 remain unassigned.

It is important to point out that the rate at which 4B symbols are conveyed via the MAC/PLS matches that of the 5B symbols' conveyance via the channel. In other words, the 4B symbols are clocked at a rate of 2.5 Mb/s, while the PMA is handling DME bits at 12.5 Mb/s, thus a DME-encoded 5B symbol is transmitted by the PMA in the same amount of time (nominally 400 ns) as a 4B nibble is received at MAC/PLS. This relevant property of the system is visualized in Figure 4.4.

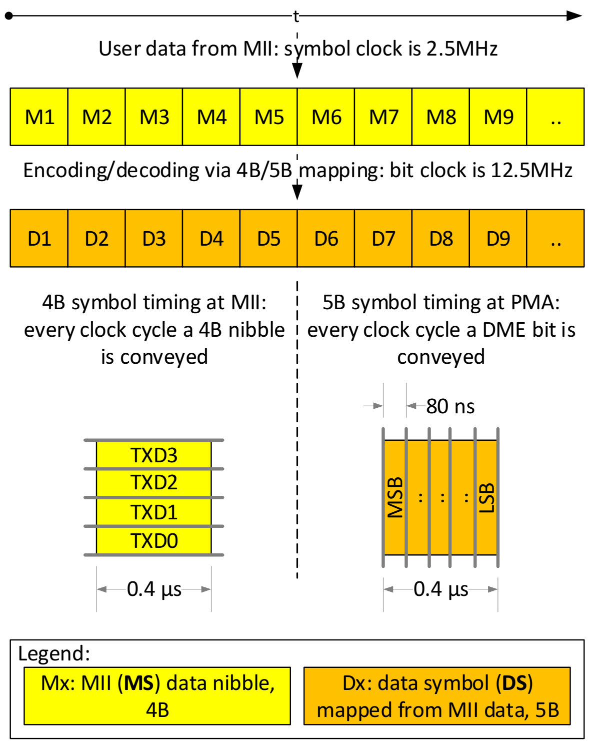

t
User data from MII: symbol clock is 2.5MHz
M1 M2 M3 M4 M5 M6 M7 M8 M9 ..
Encoding/decoding via 4B/5B mapping: bit clock is 12.5MHz
D1 D2 D3 D4 D5 D6 D7 D8 D9 ..
4B symbol timing at MII: every clock cycle a 4B nibble is conveyed
5B symbol timing at PMA: every clock cycle a DME bit is conveyed
80 ns
TXD3
TXD2
TXD1
TXD0
MSB
LSB
0.4 µs
0.4 µs
Legend:
Mx: MII (MS) data nibble, 4B
Dx: data symbol (DS) mapped from MII data, 5B

Figure 4.4: Timing behind the 4B/5B mapping of 10BASE-T1S

4.5 The underlying Error Correcting Code (ECC)

Let us define a set of notions that will aid further discussions:

- A **linear block code** is characterized by an (n, k, d_{min}) triplet, where n stands for the number of codeword symbols, k denotes that of message symbols, and d_{min} represents the **minimum distance** of the code [76];

- **Maximum Distance Separable (MDS)** codes are linear block codes such, that $d_{\text{min}} = n - k + 1$ [77]. In other words MDS codes are $(n, k, n - k + 1)$, often referred to simply as (n, k), and $t = \lfloor (n - k)/2 \rfloor$, where t denotes the maximum number of correctable errors. As described in Section 4.3.2.B, the available redundancy to implement an FEC is relatively small, and therefore the MDS codes that use the code space efficiently are the preferred choice;

- A **systematic code** is an error correcting code in the codeword of which the original encoder input (user) data appear, typically – but not necessarily – aligned to the beginning of the codeword.

The PHY architecture provides a sequence of synchronized bits along the receive path, thus the underlying ECC has to operate on binary information, or some natural extension of that.

In order to be able to deal with FSs in the output of the encoder, systematic ECC is a must. Moreover, because the Ethernet frame is variable size, the codeword of the ECC should be as short as possible.

Due to this specific set of requirements, many of the ECC-candidates – for example the Bose-Chaudhuri-Hocquenghem (BCH), the convolutional or the Low-Density Parity-Check (LDPC) codes – would not be appropriate and the selection points into the direction of a systematic MDS linear block code with known, low-complexity implementations.

Forward error correction based on linear block codes has ample history within the scope of Ethernet [2, 78, 79]. Choosing a method that has known implementations in this area would not only be an option, but a preferred choice.

Additionally, the low complexity requirements explained in Section 2.2 and in Section 4.3.2.C require a solution-candidate that has existing silicon implementation with low gate count that is compatible with an underlying binary modulations, such as DME.

It is known [80], that the only MDS binary codes that exist are the trivial (n, n), the repetition $(n, 1)$, and the Single Parity Check (SPC) $(n, n-1)$ codes, therefore the ECC of choice would need to operate on a Galois Field (GF).

ECCs that work over GF are specific to the prime number that forms the field or – if a power of the prime is used – the extended field. For digital technology, the most well-known, and inherently compatible solutions, work over an extended field GF(2^n), however an implementation may very well choose a non-binary field, such as GF(3^n). Our research involved running a survey within Institute of Electrical and Electronics Engineers (IEEE) 802.3, as well contacting silicon vendors who provide reusable FEC IP blocks. The result of this effort showed that while there is a multitude of proven FEC solutions using linear block codes, and the industry as well as Ethernet have ample experience with these, all of those work either over binary, or binary extended fields. In other words, practical implementations of FEC schemes using ternary or larger base fields were not found, hence our research also settled on codes over binary extended fields.

4.5.1 Reed-Solomon Error and Erasure Correcting Codes

RSCs belong to a family of BCH codes [81, 82] that meet all the requirements explained in Section 4.5. This group of linear cyclic MDS block codes over a finite field (GF) can be constructed in both systematic and non-systematic mode.

RSCs also have the following additional advantages:

- A coding technique called **codeword shortening** allows unique tailoring of the code performance, and codeword truncation to improve overall performance [83];
- Encoders and decoders for known code parameters can be considerably optimized theoretically, as well as in digital silicon technology;
- A multitude of efficient and proven silicon implementations exist for extended fields where the base prime is 2 [84];
- As explained in Section 4.1.1, if Physical Medium Attachment Sublayer's Receive Function (PMA_RX) is capable of signaling erasure, RSC is perfectly capable of providing **erasure correction** as well, for these cases doubling the performance of the scheme.

For all these reasons RSCs are a perfect fit for the problem at hand, subject to the constraint that a method can be found to avoid appearance of an FS anywhere in the RSCs codeword.

In conclusion, our research has narrowed down on Reed-Solomon Error and Erasure Correcting Codes (RSCs) [85] over the extended binary field GF(2^n) as the underlying ECC.

4.5.2 Field size vs. interleaving and other code parameters

As explained earlier, one of the few key attributes of an RSC is the GF it is defined over. Actual FEC implementations are heavily optimized around the parameters that define them [86] and rely on pre-calculated lookup tables to considerably speed up arithmetic operations between field elements (scalars) and polynomials over these. As the size of such lookup tables scales quadratically with the size of the GF, it is essential to select a field of small size, to keep the silicon count at bay.

According to the explanation in Section 4.2, the target of this theme of our research has been proposing a burst error correcting scheme, however RSC is capable of correcting at least t symbol errors anywhere in the codeword. To have the correcting capability per-frame configurable, either the code generator polynomial would need to be variable, or other – more complex – techniques would need to be used, making implementations overly difficult.

Another direction is using interleaving in conjunction with a single underlying RSC. Interleaving is a well-known technique, also commonly applied in the Ethernet world [87], that allows tuning the burst error and erasure correcting capabilities of the code by changing the depth of interleaving. If interleaving is used however, a lower bound on the size of GF has to be maintained, as if the number of bits necessary to represent all field elements is not divisible by $\log_2(32) = 5$, then interleaving does not achieve the increase in these capabilities. Because 10BASE-T1S uses 5B symbols, it follows that the smallest extended field to be considered for the FEC is GF(2^5), a.k.a. GF(32).
It is known [84] that over GF(32) an (n, k) RSC exists, such that $n \leq (32 - 1 = 31)$, and code shortening makes it possible to choose any integer in $[1, n - d_{\min} + 1]$ as k. This method works as follows:

1. The unused input symbols of the RSC encoder are replaced by a pre-agreed *constant 5B pattern*. As the scheme is constructed to be systematic, these codewords also appear on the output of the encoder;
2. The constant pattern is truncated from – typically the middle of – the RSC codeword;
3. When such truncated codeword is received, it gets padded by receiver with the pre-agreed pattern, before being fed to the RSC decoder.

In conjunction with this process, the following two things are worth noting:

- The extent (length) of code shortening is not conveyed in the codeword, thus pre-agreement between the transmitter and the receiver(s) must be in place, or other techniques – outside of the scope of the RSC shown here – must be present to relay such information for each frame;
- The *constant 5B pattern* may be anything, but is typically set to be a sequence of zero bits, as – due to the nature of the arithmetic over the binary field – this allows the encoding and decoding processes to completely skip doing calculations on the truncated part [86];

Throughout the dissertation we will refer to this RSC through its parameters $(n, n - 2t)$.

4.6 The proposed FEC scheme

Throughout this section, we will present explanation, concepts, constructs, and results under the assumption that $t = 1$. This is not to limit the scope of the research, and later – in Section 4.6.5.A – explanation is provided on why solution under the proposed scheme for $t > 1$ may not exist.

4.6.1 Encoding process without considering FSs

To summarize previous sections, we can state that an $(n, n - 2t)$ RSC over GF(32) may be applicable to the problem. Based on this, an encoding process would proceed as follows:

1. First some 4B user data nibbles from the MII would be scrambled;
2. Next, the output of the scrambler would be **concatenated** to directly form 5B symbols, without applying the standard 4B/5B mapping shown in Table 4.2;
3. After having formed k 5B symbols, these would be fed to the RSC encoder, producing a codeword that consists of n 5B symbols;
4. Finally, the encoded block is passed to the PMA for DME-encoded serial transmission over the wire according to Clause 147.

Before going further with our observations, let us introduce the following 2 new parameters:

- Let c denote the total number of 5B symbols in the RSC codeword, including the parity check symbols, thus $c = n = k + 2t$;
- Let u represent the total number of concatenated 4B user data symbols (each subsequently referred to as **MS**), thus $u \leq \lfloor 5k/4 \rfloor$.

There are at least 3 things worth noting here:

- This process created an FEC codeword that consists of the following parts:
  - 5B **D**ata **S**ymbols (**DS**s): These are formed by the concatenation of u scrambled **MS**s;
  - 5B **P**arity **S**ymbols (**PS**s): $2t$ 5B symbols created by the RSC encoder;
  - **S**ignaling **B**its (**SB**s): These are the remaining bits in the codeword that have not been occupied by a **DS** or a **PS**.

    The name originates from the fact, that **SB**s are to be assigned the important role of carrying signaling information for the proposed FEC scheme.
- According to the timing details shown in Figure 4.4, the FEC codeword created through this process has the exact same length in time domain as the **MS**s that produced the RSC codeword;
- The **SB**s are "free bits" and – as to be shown in Section 4.6 – these will be used to avoid the appearance of FSs anywhere in the RSC codeword.

Subsequently we will refer to such a construct and its extensions as a $\{c, u\}$ **coding scheme**. An example of such a scheme is show in details in Figure 4.5 [(60)] for $c = 15$ and $u = 15$.

[(60)] To maintain clarity, scrambling not shown in these figures.

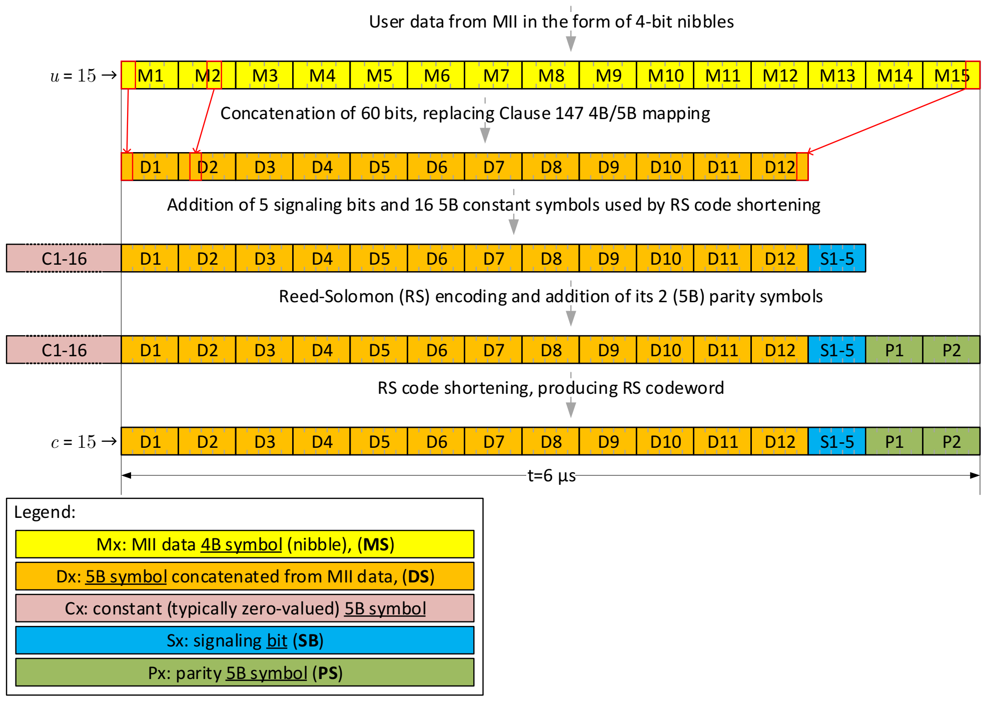

User data from MII in the form of 4-bit nibbles
u = 15 →
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
M11
M12
M13
M14
M15
Concatenation of 60 bits, replacing Clause 147 4B/5B mapping
D1
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
Addition of 5 signaling bits and 16 5B constant symbols used by RS code shortening
C1-16
S1-5
Reed-Solomon (RS) encoding and addition of its 2 (5B) parity symbols
P1
P2
RS code shortening, producing RS codeword
c = 15 →
t=6 µs
Legend:
Mx: MII data 4B symbol (nibble), (MS)
Dx: 5B symbol concatenated from MII data, (DS)
Cx: constant (typically zero-valued) 5B symbol
Sx: signaling bit (SB)
Px: parity 5B symbol (PS)

Figure 4.5: A possible encoding process of a $\{15, 15\}$ coding scheme, based on a $(15, 13)$ RSC over GF(32), when $t = 1$

It is worth noting that Figure 4.5 does show the additional step necessary for code shortening. The 16 zero-valued 5B symbols denoted by C1–C16 are added, then removed by the encoding and decoding processes (respectively), according to the explanation in Section 4.5.2. For simplicity's sake, in the later parts of the dissertation, we will omit showing these steps.

When $(4u \bmod 5) \neq 0$, there is a 5B symbol in each codeword that holds mixed information from and **MS** and from **SB**s. For clarity, let $d = \lfloor 4u/5 \rfloor$ denote the number of **complete DS**s where no such mixing of information occurs. An example of this for $\{19, 19\}$ is shown in Figure 4.6, where $d = 15$.

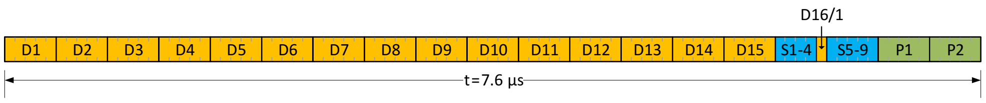


Figure 4.6: A possible layout of the $(19, 17)$ RSC codeword behind a $\{19, 19\}$ coding scheme, when $t = 1$

An actual implementation may choose an arbitrary ordering and arrangement of the inner parts of the codeword, denoted by **DS**, and **SB**, and **PS**, however for the sake of consistency and simplicity, in the dissertation we will use only the order displayed in Figure 4.5 and Figure 4.6. The reason for this is that it follows the "Least Significant Bit (LSB) first" principle widely used in Clause 147 and it also allows pre-coding.

4.6.2 Need for an additional FS (FECESD)

The method described in Section 4.6.1 is capable of exposing the unused redundancy present in the 4B/5B mapping to implement error-resilience, however it does not offer a way to signal end-of-frame (ESD) in such a manner. Relying purely on the ESD mechanism defined by Clause 147, shown in Figure 3.1, is not sufficient, as corruption of any parts of the 2 terminal 5B symbols `T` and `R` would make the end-of-frame detection fail, eliminating the capability of any FEC-enabled PHY to successfully decode the frame, even in the presence of an FEC on the payload that is otherwise well capable of dealing with noise of such magnitude elsewhere in the frame.
It is a natural requirement, that the error-resilience of the ESD provided by the FEC be at least as good as that of the payload. To be able to achieve this, we propose to reserve an additional 5B symbol to fulfil the role of FECESD. The mechanism would work as follows:

1. When the MAC/PLS ceases conveying **MS**s the FEC first pads the concatenated 4B symbols up till the next 5B symbol boundary, then immediately inserts the FECESD 5B symbol;

2. Now, the next 5B symbol is used to indicate the presence or absence of padding used in the 5B symbol preceding FECESD. This is necessary, as for example in case FECESD is inserted at `D5`, the decoder would otherwise be unable to decide whether the concatenation terminated after 4 or 5 **MS**s.

   If conveying additional information – for example good or bad ESD – is needed, the very same 5B symbol may also be used for this purpose. In this case, given that the 5B symbol following FECESD may have any of the base admissible values, this technique allows conveying additional information as well;

3. The final padding to the codeword(s) and to the superblock are applied as necessary;

4. Finally, according to the explanation in Section 4.3.3.B, the fixed 5B symbol sequence of `T` followed by a `K` is inserted to force all MACs above 10BASE-T1S PHYs to discard the received frame, to maintain backward-compatibility.

As per this, the **MS** concatenation process explained in Section 4.6.1 must not produce any **DS**s that take the value of the 5B symbol used by FECESD, as it would cause an incorrect detection of the ESD. Because of this, this 5B symbol must be treated as a 4$^{\text{th}}$ FS, in addition to the 3 base FSs introduced in Section 4.3.4. Because the 5B symbol assigned to represent FECESD may be any of the yet unassigned 5B symbols, we will subsequently refer to it as the pseudo-5B symbol of `X`.

Introducing 4$^{\text{th}}$ FS further decreases the remaining (unused) redundancy present in the 4B/5B mapping to the value of $\log_2(32-4)-4 \approx 0.81$ bits per 5B symbol mentioned in Section 4.1.2.

It is worth mentioning that FECESD has a limited scope, as it identifies the location of the last valid **MS** in the codeword, thus it must be avoided to be formed only by the concatenation process producing **DS**s consisting of only **MS**s. In other words `X` is treated as an FS only among the first d **DS**s – so, among all the complete **DS**s – in each codeword.

Subsequently, the term FS will be used to collectively refer to the actual 5B symbols (FSs) that need to be avoided in the actual part of the codeword, however when distinction must be made to avoid ambiguity, the following terminology will be used:

- The 3 FSs introduced in Section 4.3.4 that must not appear anywhere in the codeword will be referred to as a **base FSs**, denoted by $\mathcal{F}_\text{b} = \{\texttt{T}, \texttt{R}, \texttt{I}\}$;
- The base FSs together with the 5B symbol that represents FECESD ($\texttt{X}$) will be referred to as a **payload FSs**, denoted by $\mathcal{F}_\text{p} = \mathcal{F}_\text{b} \cup \{\texttt{X}\}$.

All the details of the FSs are summarized in Table 4.3.

Table 4.3: Details of the Forbidden Symbols (FSs)

In Clause 147 (4B/5B)			
Symbol name	Special function	Binary value	Index and polynomial forms
T	ESD, HB	01101	$\alpha^8 = \alpha^3 + \alpha^2 + 1$
R	ESDOK, ESDBRS	00111	$\alpha^{11} = \alpha^2 + \alpha + 1$
I	SILENCE	11111	$\alpha^{15} = \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$
Not in Clause 147			
X	FECESD	*(as applicable)*	

Note: the exponential forms are shown for the example case where the field generator polynomial $p(x) = x^5 + x^2 + 1$ over GF(2^5), where α is the primitive element (root) of the field

4.6.3 Avoiding Forbidden Symbols in the codeword

Section 4.6 hitherto introduced a construction technique that allows creating various $\{c, u\}$ coding schemes that are to meet the backward-compatibility requirements described in Section 4.3 with the exception of avoiding appearance of FSs in the RSC codeword. For this reason we refer to these $\{c, u\}$ coding schemes only as candidates, since they are not yet complete solutions.

The remaining parts of this chapter fill this gap in the solution by proposing a set of novel techniques and appropriate combination of those, in a way that guarantees avoiding the appearance of any FSs in all 3 distinct parts of the codeword introduced in Section 4.6.1.

4.6.3.A Avoiding the appearance of an FS among DSs

The 5B symbols referred to as **DS**s are formed by the direct concatenation of the unconstrained (free) user input data from the MAC conveyed via the Media Independent Interface (MII), as shown in Figure 4.5, therefore no restrictive assumption can be made with regards to their values.

Note that in this section we are discussing only complete **DS**s, and the solution for the case when in a 5B codeword symbol, bits from both an **MS** and **SB**s get bundled – for example in the case of the 16th codeword symbol in the $\{19, 19\}$ coding scheme shown in Figure 4.6 – will be discussed separately in Section 4.6.3.C.

Let $\mathcal{A}_\text{p} = \text{GF}(32)\backslash\mathcal{F}_\text{p}$ denote the set of admissible payload 5B symbols. To eliminate any FS that may appear among the **DS**s, we propose the following coding technique:

1. Let us define a forward-pointing **linked-list** that connects all complete **DS** that happen to have become a payload FS during the concatenation process of **MS**s (if any);

2. During this process, let us **transcode** each complete **DS** x such that $x \in \mathcal{F}_\text{p}$, to a 5B symbol y such that $y \in \mathcal{A}_\text{p}$ via an invertible mapping, in a way that each y represents x, and it also contributes to representing the linked-list. This process makes the size of the output independent from the number of FSs.

Following this construct, the process of eliminating all FS among **DS**s would go according to the following sequence:

1. Record the index (position) of the **first** payload FS among the **DS**s. If there are none present, use a value that represents immediate End of List (EoL);

2. Transcode each **remaining** payload FS in their order of appearance, so that 5B symbol value replacing each FS would be an admissible symbol encoding both of the following facts in conjunction:

   - f: information on which actual FS has been transcoded. As – according to Section 4.6.2 – there are 4 payload FSs that may appear among **DS**s, this requires at least two bits of information;
   - δ: the distance to the next immediate FS, or EoL if there are no more payload FSs, among the complete **DS**s.

   The invertible mapping used by the transcoding is an implementation detail, however one possible solution is later shown in Table 4.8 on page 74;

3. The first two steps are repeated until all FSs are eliminated by the transcoding.

Some simple examples of this process for $d = 15$ (used for example by a $\{19, 19\}$ coding scheme) are shown in Figure 4.7.

To see why this approach is still not complete, let us make the following three observations:

- An unconstrained 5B symbol can represent 32 distinct values, which has to encode both of the independent values of f and δ at the same time;

- The four payload FSs directly decrease the number of admissible symbols from 32 to 28;

- As mentioned in the definition of the process, δ has to be able to encode EoL as well.

From the previous observations it is apparent that, if $\mathcal{F}_\text{p}$ denotes the set of payload FSs, the process described above would not allow δ to be larger than $\delta_\text{max} = \lfloor(|\text{GF}| - |\mathcal{F}_\text{p}|)/|\mathcal{F}_\text{p}|\rfloor - 1$, thus $\delta_\text{max} = 6$ in case of the proposed scheme. For example, if D5 and D12 happen to be FSs [61], the δ part of the admissible 5B symbol value that transcodes the FS at D5 would be unable to represent $\delta = 7$, which is the distance between itself and FS at D12.

[61] This is shown in example #6 in Figure 4.8.

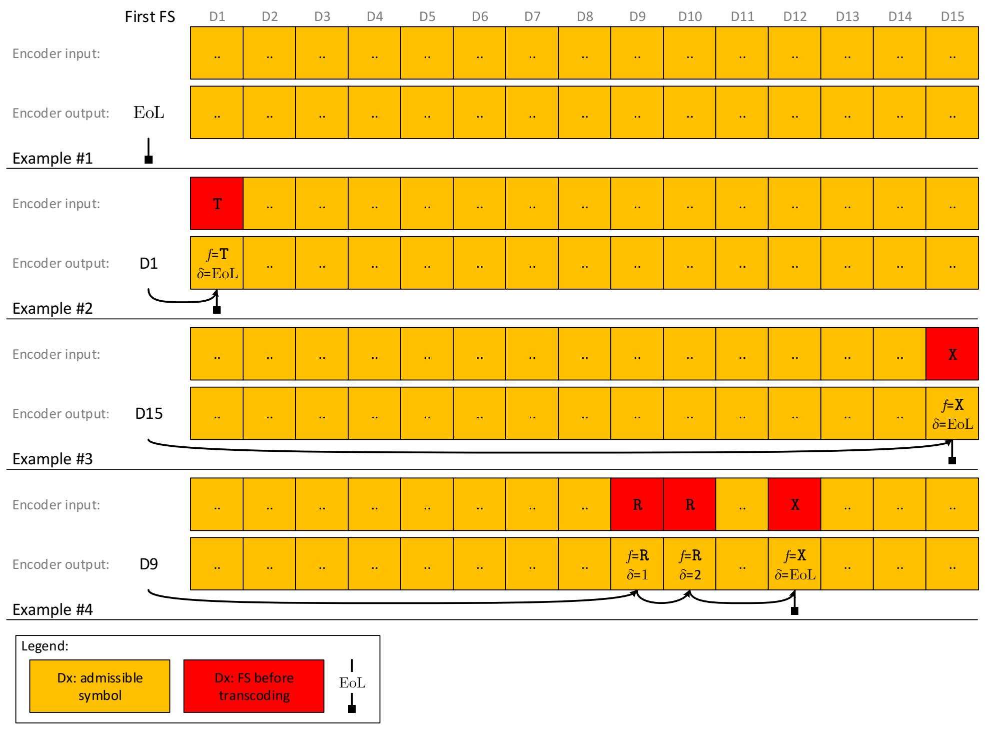


Figure 4.7: Some simple example FS-transcoding linked-lists over 15 **DS**s

We propose that this problem be resolved by the definition and application of a construct referred to as **Special Constellation Descriptor (SCD)**, which allows forming the linked-list in those cases when the actual distance between any two neighboring FSs – denoted by δ_{act} – is larger than $\delta_{\max}$. The SCD is to indicate one of the following two situations:

- Either that $\delta_{\text{act}} \leq \delta_{\max}$ for all payload FSs;
- Or that there exists at least one FS in case of which $\delta_{\text{act}} > \delta_{\max}$.

  For these FSs, also do the following:

  1. Let δ for the affected FSs be $\delta = (\delta_{\text{act}} \mod (\delta_{\max} + 1))$;
  2. Record the following two attributes for each of the affected FSs:
     - Their index position z in the linked-list;
     - And $m = \lfloor \delta_{\text{act}} / (\delta_{\max} + 1) \rfloor$.

From the process explained above, let us combine the index of the first FS among **DS**s *and* the SCD into a single value, subsequently referred to as **FS Transcoding Recipe (FTR)**. Figure 4.8 shows how the complete process works for some examples cases with $d = 15$ **DS**s, while an example mapping between the FTRs and FTR indices will later be shown in Table 4.7 on page 73.
The number of FTRs needed to implement a $\{c, u\}$ coding scheme depends on d, in other words the number of complete **DS**s in it. The closed formula representing the

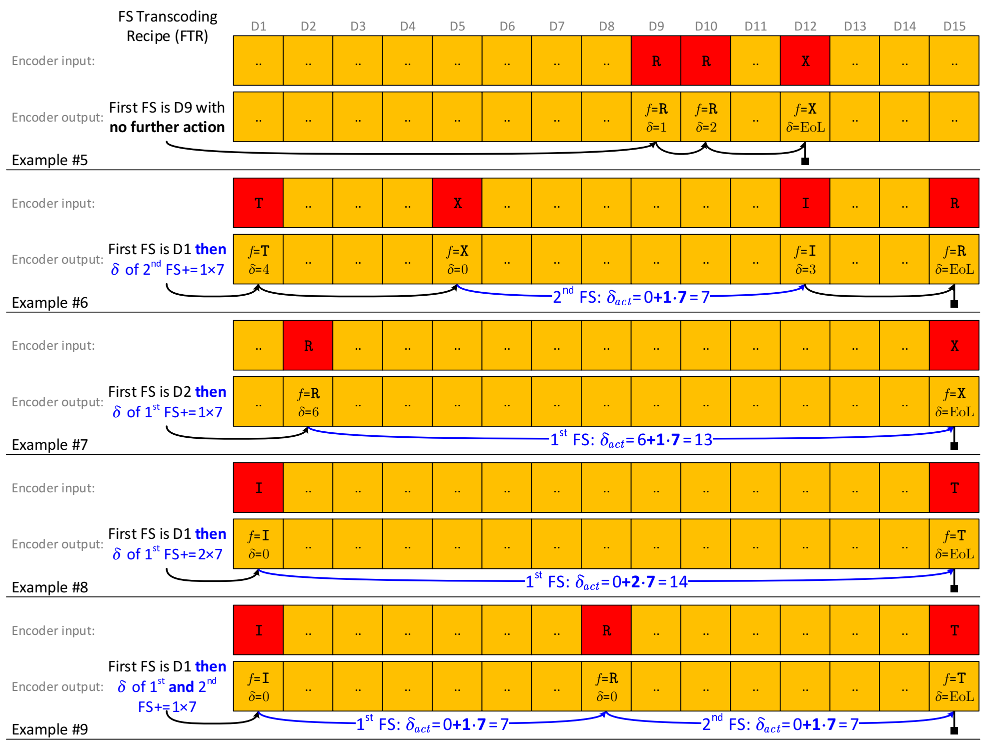

FS Transcoding Recipe (FTR)
D1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12 D13 D14 D15
Encoder input:
Encoder output:
First FS is D9 with no further action
Example #5
First FS is D1 then δ of 2nd FS+=1×7
2nd FS: δact=0+1·7=7
Example #6
First FS is D2 then δ of 1st FS+=1×7
1st FS: δact=6+1·7=13
Example #7
First FS is D1 then δ of 1st FS+=2×7
1st FS: δact=0+2·7=14
Example #8
First FS is D1 then δ of 1st and 2nd FS+=1×7
1st FS: δact=0+1·7=7
2nd FS: δact=0+1·7=7
Example #9

Figure 4.8: Some more complex example FS-transcoding linked-lists over 15 **DS**s

relationship between d and the number of needed FTRs, subsequently referred to as $G(d)$, may be obtained as follows:

1. For $1 \leq j \leq d$, let us define $e(j)$ as the maximum number of (z, m) pairs in any SCD for j complete **DS**s. It can be seen, that $e(j) = \lfloor (j-1)/(\delta_{\max}+1) \rfloor$, as this is the case when FSs happen to be exactly $\delta_{\max}+1$ steps apart [(62)];

2. Now, to obtain the number of FTRs among exactly j complete **DS**s where the first **DS** is an FS, *all forward-pointing distinct walks that consist of all possible combinations of single steps, and the integer multiples of* $(\delta_{max}+1)$*-long steps that do not end in a single-step* need to be counted. In here, a *walk* is a set of linked lists that share an SCD.

   This is because – according to the definition – a walk does require a unique SCD, while all other walks can be derived from another SCD using the (f, δ) pairs formed by transcoding, therefore the walks counted this way are all necessary, while nothing further is required, thus these are the **necessary and sufficient conditions**. This relationship is expressed by the recursive formula $F(j)$ as shown by Equation (4.1);

[(62)] For example when $d = 15$, then $e(d) = 2$ as shown in example #9 in Figure 4.8.

$$F(j) = \begin{cases} F(j-1) + \sum\limits_{i=1}^{e(j)} F(j - i(\delta_{\max}+1)) & \text{if } e(j) > 0; \\ 1 & \text{otherwise.} \end{cases} \tag{4.1}$$

3. As an FTR is defined by the first FS among **DS**s *and* the SCD, we can calculate all the FTRs needed by counting the single EoL and the sum of the quantity of all SCDs with all possible first FSs.

   In short, as d denotes the number of **DS**s, the quantity of all FTRs needed by a $\{c, u\}$ coding scheme can be expressed by $G(d)$ as shown by Equation (4.2);

$$G(d) = 1 + \sum_{j=1}^{d} F(j) \tag{4.2}$$

If we plot all meaningful values for $G(d)$, we get the curve shown in Figure 4.9. If we repeat the plot using a logarithmic vertical axis, as shown in Figure 4.10, it reveals that a $\{c, u\}$ coding scheme needs ≈ 0.35 bits of additional information at every increment of d.

4.6.3.B Avoiding the appearance of an FS among PSs

The $2t$ 5B symbols representing **PS**s are the direct product of the RSC encoder. Because the encoder's input is unconstrained and the RSC code is MDS, there exists no mapping that would avoid the appearance of FS among **PS**s for an arbitrary input. As per this, some additional, yet unused, free bits – that are external to the **PS**s – must be used to influence the 5B symbol values the **PS**s may take.
Figure 4.11 shows how the $2t = 2$ **PS**s are produced by the RSC encoder, through the iterative process of polynomial long division. If $\lambda, \omega \in \mathrm{GF}(32)$, then what is worth noting is that for a given RSC code generator polynomial of $g(x) = x^2 + \lambda x + \omega$ [63], both **PS**s are fully determined by the $2t + 1$ input symbols $2t$ before the last sub-division of the polynomial long division. These 3 values for $t = 1$ – also highlighted by a thick red caret in Figure 4.11 – are as follows:

- a and b: These are the cumulative remainders of all previous sub-divisions;
- s: the last 5B symbol formed by the **SB**s, in this case the 5 bits denoted by S5–S9.

This gives an opportunity to influence the normal RSC encoding process so that when a and b become available during the polynomial division, s would be selected from the set of admissible 5B symbol values, such that neither P1, nor P2 would end up being an FS. The direct application of this method would use the space in s efficiently, but it would require a large lookup table that allows selection of s based on exact values of both a and b. While this process may be implementable, it may not be feasible for low-complexity systems, such as 10BASE-T1M, therefore we propose making this lookup considerably simpler by relying on the following 2 notable facts:

- All 3 base FSs to be avoided among **PS**s are identical at their LSBs and the central bits, as shown in Table 4.3 in blue and red colors (respectively);

[63] Thus λ and ω represent the 2 coefficient for the code generator polynomial, which is derived from the general form of $g(x) = (x + \alpha^{\sigma})(x + \alpha^{\sigma+1})$, therefore $\lambda \neq 0$ and $\omega \neq 0$.

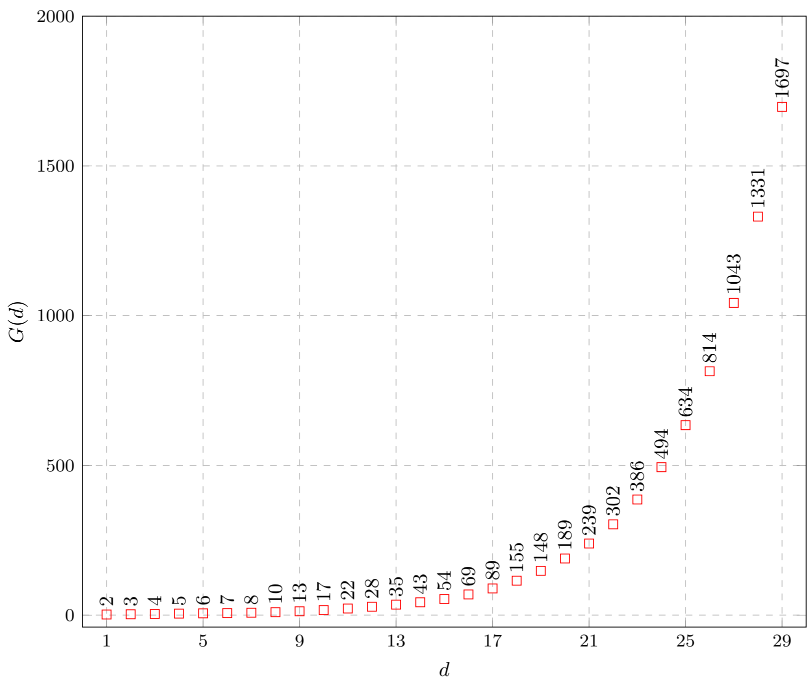


Figure 4.9: The relationship between number of complete **DS**s in the codeword (*horizontal axis*) and the number of FTRs each requires (*vertical axis*)

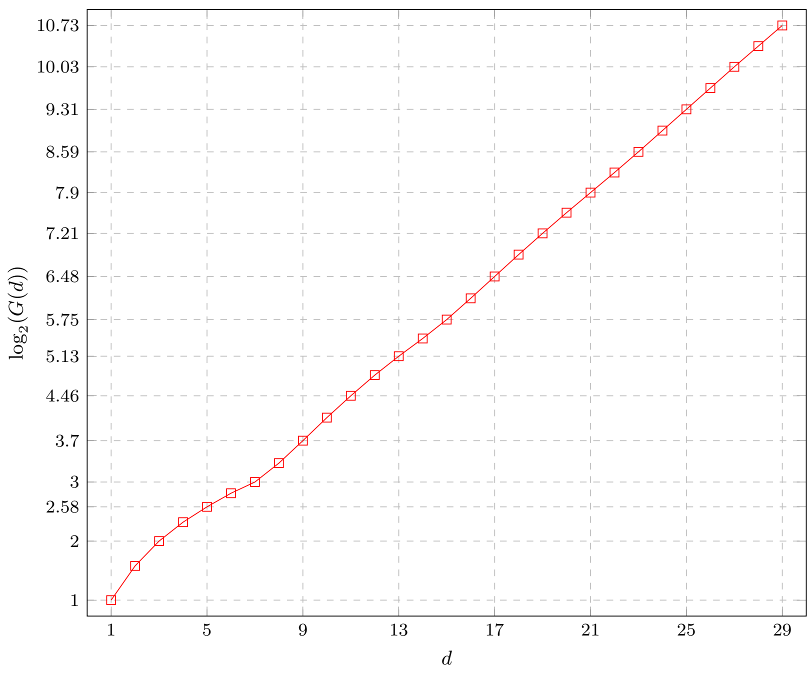


Figure 4.10: The relationship between number of complete **DS**s in the codeword (*horizontal axis*) and the quantity of information (in bits) needed to encode all FTRs (*vertical axis*)

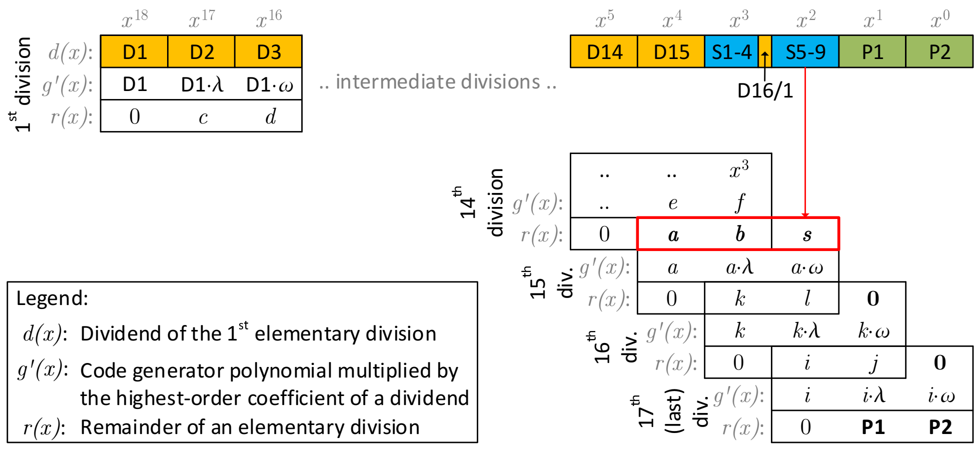

1st division
d(x):
g'(x):
r(x):
D1 D2 D3
D1 D1·λ D1·ω
0 c d
.. intermediate divisions ..
D14 D15 S1-4 S5-9 P1 P2
D16/1
14th division
15th div.
16th div.
17th (last) div.
a b s
k l
i j
P1 P2
Legend:
d(x): Dividend of the 1st elementary division
g'(x): Code generator polynomial multiplied by the highest-order coefficient of a dividend
r(x): Remainder of an elementary division

Figure 4.11: The complete polynomial long division in a $\{19, 19\}$ coding scheme that produces **PS**s P1 and P2, when $t = 1$

- Elements (scalars) of GF(2^n) are added by modulo-2 addition of the coefficients, which in binary form means bit-by-bit Exclusive Or (XOR) of the 2 elements.

If $\mathsf{P1}, \mathsf{P2}, a, b, i, j, k, l, s \in \mathrm{GF}(32)$, then – based on previous observations and according to Figure 4.11 – the 2 **PS**s (namely P1 and P2) are the result of the set of elementary equations shown in Equation (4.3) (63).

$$\begin{aligned} \mathsf{P1} &= j + i\lambda \\ \mathsf{P2} &= i\omega \\ i &= l + k\lambda \\ j &= k\omega \\ k &= b + a\lambda \\ l &= s + a\omega \end{aligned} \tag{4.3}$$

The straightforward simplification of Equation (4.3) leads to the results shown in Equation (4.5) and Equation (4.4).

$$\mathsf{P1} = \overbrace{a\lambda^3 + b(\lambda^2 + \omega)}^{\mathsf{P1}_{\mathrm{left}}} + s\lambda \tag{4.4}$$

$$\mathsf{P2} = \overbrace{a(\lambda^2\omega + \omega^2) + b\lambda\omega}^{\mathsf{P2}_{\mathrm{left}}} + s\omega \tag{4.5}$$

What is worth recognizing here is that – due to the nature of XOR – P1 is fully determined by $s\lambda$, while P2 is fully determined by $s\omega$, therefore to avoid FSs from appearing among **PS**s, the $\{c, u\}$ coding scheme may proceed as follows:

1. Choose any bit location in the 5B symbol, where all FSs have matching values: As explained above, the LSB and the central bit positions meet this criterion. While either of these may be used to achieve the desired result, in our demonstrations we will use LSB for this purpose;

2. Given the set of admissible base 5B symbols $\mathcal{A}_\text{b} = \text{GF}(32) \setminus \mathcal{F}_\text{b}$, and bit $v \in \{0, 1\}$, let us define the set-mappings S_1 and S_2 of $\mathcal{A}_\text{b}$ according to Equation (4.7) and Equation (4.6), where the function LSB_of(n) returns the value of the LSB of n [(64)];

$$S_1(v) = \{x \in \mathcal{A}_\text{b} \mid \text{LSB\_of}(x\lambda) = v\} \tag{4.6}$$

$$S_2(v) = \{x \in \mathcal{A}_\text{b} \mid \text{LSB\_of}(x\omega) = v\} \tag{4.7}$$

3. During the polynomial division shown in Figure 4.11, just after a and b become available, calculate $\mathsf{P1}_\text{left}$ and $\mathsf{P2}_\text{left}$ using Equation (4.5) and Equation (4.4) (respectively);

4. Finally, pick a single value for s such, that Equation (4.8) is satisfied.

$$s \in S_1(\text{LSB\_of}(\mathsf{P1}_\text{left})) \cap S_2(\text{LSB\_of}(\mathsf{P2}_\text{left})) \tag{4.8}$$

As this method guarantees that both LSB_of($\mathsf{P1}_\text{left} + s\lambda$) and LSB_of($\mathsf{P2}_\text{left} + s\omega$) will always be zero, it follows that neither P1, nor P2 may turn out to be any of the 3 base FSs. $\square$

While the clear advantage of this method is that it avoids the appearance of any of the base FSs among the **PS**s without (or with very small) lookup tables, it effectively avoids more than those 3 values: with this method all **PS**s will have the value 0 at their LSBs.

4.6.3.C Avoiding SBs forming an FS

As explained in Section 4.6.1, **SB**s are “free bits”, the purpose of which in the $\{c, u\}$ coding scheme is to support avoiding FSs among **DS**s and **PS**s, while making sure 5B symbols where **SB**s appear do not form any of the base FSs either.

As explained in Section 4.6.1, in the proposed construct, **SB**s make up one or more complete codeword symbols, and when $(4u \mod 5) \neq 0$ those also form a mixed 5B symbol in each codeword, where bits also from an **MS** appear. For example, as shown in Figure 4.6, in case of a $\{19, 19\}$ coding scheme S5–S9 form a complete codeword symbol, while S1–S4 share a 5B symbol with D16/1.

In a $\{c, u\}$ coding scheme, when a codeword symbol has any **SB**s in it, the responsibility of avoiding an FS from being formed by the 5B symbol falls on those **SB**s. The limits of information representation (signaling) capabilities of these 5B symbols depends on the number of **SB**s in them, which can be maximized using the allocation method shown in Table 4.4.

Care must be taken that the 5B symbol denoted by s introduced in Section 4.6.3.B is itself an admissible codeword symbol made up of only **SB**s, and while Table 4.4 suggests s should be able to encode 29 distinct signaling values, this is not the case, as s also contributes to the FS avoidance for the **PS**s. For this reason, special attention has to be paid with respect to the actual signaling capabilities of s. The method described in Section 4.6.3.B defined the set-mappings S_1 and S_2 of admissible symbols.

(64) As $S_1(0) \cup S_1(1) = \mathcal{A}_\text{b}$ and $S_1(0) \cap S_1(1) = \emptyset$, and S_2 works similarly, these two set-mappings partition $\mathcal{A}_\text{b}$.

Table 4.4: Relationship between number of **SB**s (*in bits*) in a 5B symbol, and the number of signaling values it can encode without forming an FS

Number of		A possible method of construction that maximizes the signaling capabilities of each 5B symbol with any **SB**s in it
SBs in the 5B symbol	signaling values	
1	1	LSB or the central bit must be picked, and it needs to be 0: $2^1 - 1 = 1$
2	3	Assign LSB and the central bit, and avoid using 1-1: $2^2 - 1 = 3$
3	6	Same as above, then pick any 3rd bit: $2(2^2 - 1) = 6$
4	13	Pick any 4 bits and avoid the 3 patterns FSs have there: $2^4 - 3 = 13$
5	29	Simply avoid using the 3 FSs: $2^5 - 3 = 29$

It can be seen, that the signaling capabilities of s is upper-bounded by s_{sig} defined by Equation (4.9).

$$s_{\text{sig}} = \min_{b_1 \in \{0,1\}} \min_{b_2 \in \{0,1\}} |S_1(b_1) \cap S_2(b_2)| \tag{4.9}$$

As visible, the actual value provided by Equation (4.9) depends on where the base FSs happen to fall in the intersections of these sets, which depends on both the actual field and code generator polynomials $p(x)$ and $g(x)$ (respectively) chosen by an implementation. While an actual value for s_{sig} will later be shown in Section 4.6.6, lower and upper limits on the possible values of s_{sig} may be determined through the following thought experiment, under the obvious assumption that $\lambda \neq \omega$ [(65)]:

1. Multiplication of all elements of GF with a non-zero scalar is a **bijective** mapping, thus if the two set-mappings S_1 and S_1 had been defined not to exclude any FSs, each of the 4 intersections of the sets formed through these modified set-mappings would have exactly $|\text{GF}(32)|/4 = 8$ elements, in other words s_{sig} would always be 8;

2. Now, if the return to the original definitions for S_1 and S_2 (in other words FSs do get excluded), it can be observed that – depending on $p(x)$ and $g(x)$ – at least 1 FS will be excluded from at least 1 of the 4 intersections of the sets, while at most all – the 3 – base FSs may be excluded from any of the 4 intersections;

3. This leads to the conclusion that in the proposed scheme s_{sig} is in $[((|\text{GF}(32)|/4) - 1), ((|\text{GF}(32)|/4) - |\mathcal{F}_{\text{b}}|)]$, thus $5 \leq s_{\text{sig}} \leq 7$. $\square$

Let ℓ denote the number of **SB**s (in bits), thus $\ell = 5(c - 2t) - 4u$. Now, to summarize, it can be stated that **SB**s are used by the scheme to encode information necessary to avoid the appearance of FSs among **DS**s and **PS**s. The latter is achieved by forming the last 5B symbol (s) comprised of **SB**s as described in Section 4.6.3.B, the process of which also leaves at most 7 values available for additional signaling.

(65) If $\lambda = \omega$, then $S_1(v) = S_2(v)$, thus $S_1(0) \cap S_2(1) = S_1(1) \cap S_2(0) = \emptyset$, therefore $s_{\text{sig}} = 0$, which would not be a useful construct for the proposed scheme.

FS avoidance among **DS**s is achieved by encoding the FTRs defined in Section 4.6.3.A by these 7 values, plus the signaling capabilities of the remaining $\ell - 5$ **SB**s.

Let $h(i)$ denote the number of signaling values carried by i bits in a 5B symbol according to Table 4.4, formally shown by Equation (4.10).

$$h(i) = \begin{cases} 29 & \text{if } i = 5; \\ 13 & \text{if } i = 4; \\ 6 & \text{if } i = 3; \\ 3 & \text{if } i = 2; \\ 1 & \text{otherwise.} \end{cases} \tag{4.10}$$

Accordingly, if $\ell \geq 5$, the capability of this scheme to encode FTRs, denoted by $H(\ell)$, is expressed by Equation (4.11).

$$H(\ell) = s_{\text{sig}} \prod_{i=1}^{\lfloor \ell/5 \rfloor} h(\min(\ell - 5i, 5)) \tag{4.11}$$

4.6.4 Existence of $\{c, u\}$ coding schemes using FTR

From the previous parts of Section 4.6 it is clear how to construct $\{c, u\}$ coding schemes that are capable of avoiding FSs in any parts of the codeword, as long as $t = 1$. Until now however, c and u have been treated as free parameters, without respect to constraints presented by constructability and those of the backward-compatibility requirements defined in Section 4.3.

Let us formalize all the requirements meeting which would allow a $\{c, u\}$ coding scheme candidate to exist, as follows:

1. First, the rate r of the code, defined as $r = 4u/5c$, must satisfy $r \geq 0.8$ to meet the backward-compatibility requirement defined in Section 4.3.2.B, thus Equation (4.12) must be met by the code parameters;

$$u \geq c \tag{4.12}$$

2. Next, the number of bits represented by **MS**s must fit into the **DS**s present in the RSC codeword, and – as defined in Section 4.6.3.C – it must be possible to form s, thus Equation (4.13) must hold;

$$l \geq 5 \tag{4.13}$$

3. Finally, ℓ **SB**s must be capable of encoding all the FTRs needed by the d complete **DS**s, which is expressed by Equation (4.14).

$$H(\ell) \geq G(d) \tag{4.14}$$

Solving this system of inequalities representing existence criteria for $\{c, u\}$ coding scheme candidates for $t = 1$ and $s_{\text{sig}} = 7$ [(66)] provides the complete set of existing schemes listed in Table 4.5.

Table 4.5: Existence of all $\{c, u\}$ coding schemes, when $t = 1$

	FTRs			FTRs	
$\{c, u\}$	available	needed	$\{c, u\}$	available	needed
$\{19, 19\}$	91	54	$\{27, 27\}$	17661	239
$\{20, 20\}$	203	69	$\{27, 28\}$	1218	303
$\{21, 21\}$			$\{28, 28\}$	35322	
$\{22, 22\}$	609	89	$\{28, 29\}$	2639	386
$\{23, 23\}$	1218	115	$\{29, 29\}$	76531	
$\{24, 24\}$	2639	148	$\{29, 30\}$	5887	494
$\{24, 25\}$	203	189	$\{30, 31\}$		
$\{25, 26\}$			$\{30, 30\}$	170723	
$\{25, 25\}$	5887		$\{31, 31\}$		
$\{26, 26\}$			$\{31, 32\}$	17661	634
$\{26, 27\}$	609	239	$\{31, 33\}$	1218	814

The value of $G(d)$ for a $\{c, u\}$ coding scheme directly influences the complexity of its implementation, and – as shown in Figure 4.10 – $G(d)$ scales with d exponentially. Moreover, the larger parameters c and u get, the smaller the relative error correcting capabilities – expressed by t/u – the coding scheme provides. Already for these two reasons alone it is obvious that c and u should be chosen to be as small as possible.

From these, and the explanation provided in Section 4.5, it follows, that under the assumptions and using the techniques presented in the dissertation a $\{19, 19\}$ coding scheme is **optimal** for$t = 1$, both from practical (engineering) and theoretical perspectives.

For other reasons one might choose to use a more complex implementable coding scheme. While these diverge from the above-mentioned optimality, in return such a choice would allow additional information to be encoded into the **SB**s.

For example, by looking at Table 4.5, it becomes visible that a $\{20, 20\}$ coding scheme would allow up to $29{\cdot}7 = 203$ possible FTRs to exist, while only 69 of those would be required to avoid FSs. This would allow $\lfloor \log_2(203/69) \rfloor = 1$ free extra bit worth of information to be available to convey arbitrary data in every RSC codeword, at the cost of decreasing the relative error correcting capability of the scheme.

[(66)] $s_{\text{sig}} = 7$ is achieved for example in case of the field generator polynomial of $p(x) = x^5 + x^2 + 1$ and code generator polynomial of $g(x) = x^2 + 25x + 31$ ($\sigma = 7$).

4.6.5 Extensions to the proposed scheme

4.6.5.A On Error Correcting Codes (ECCs) with $t > 1$

Until this section, we have discussed applicability of $\{c, u\}$ coding schemes over GF(32) under the assumption that $t = 1$. This may have looked as an unnecessarily restriction, given that should any $t > 1$ schemes exist, those may provide considerable improvement to the relative error correcting capabilities. In this section we focus on this yet unexplored area of the solution candidate space, and show that **no** $\{c, u\}$ **coding schemes exist for** $t > 1$.

To achieve this, we proceed as follows:

1. It is clear that such a scheme would need to satisfy the inequalities shown in Equation (4.12) and Equation (4.13) presented in Section 4.6.4;

2. It is also apparent from the explanation in Section 4.6.3.A that the output of Equation (4.2) – that defines $G(d)$ – does not directly depend on t;

3. From the explanation in Section 4.6.3.B it is visible however, that the output of Equation (4.11) – that defines $H(\ell)$ – does depend on t.

   To alleviate for the need of solving the equations for each **PS** for $t > 1$, let us make a relevant observation: from the description in Section 4.6.3.B it is visible, that s_{sig} defined by Equation (4.9) is also an upper bound in case of $t > 1$.

   Because of this, any $\{c, u\}$ coding scheme excluded by the solution of Equation (4.12) and Equation (4.13) for $t > 1$, and that of Equation (4.14) for $t = 1$, certainly does not exist. □

The output of solving the inequalities for all values of t is visualised in Figure 4.12.

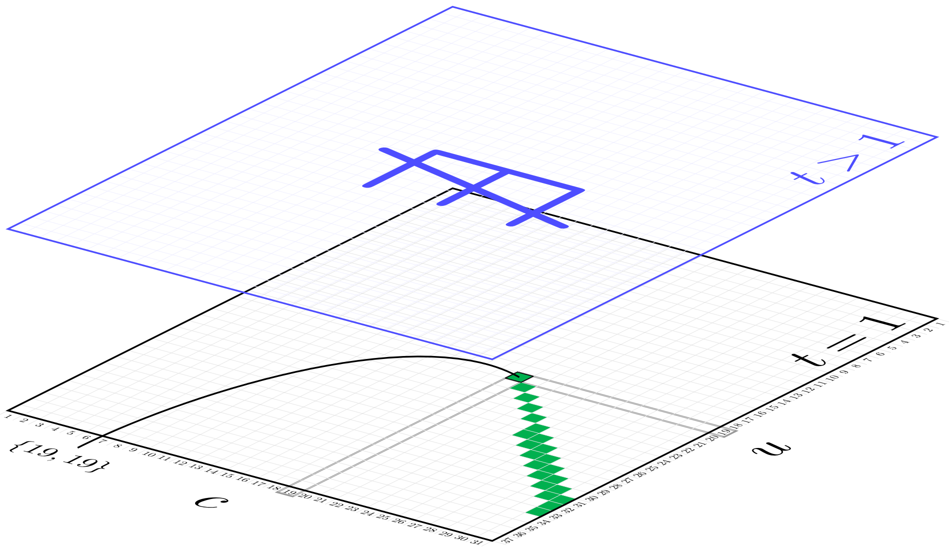


Figure 4.12: Existence of all $\{c, u\}$ coding schemes for all values of t (*green squares indicate existing schemes*)

4.6.5.B Erasure correction

Erasure detection can rely on optional side-channel information provided by the PMA_RX of a receiver, whenever certain parameters of the received digital or analog signals are outside of the valid range. For example, erasure may be assumed when the signal swing [(29)], or the timing of the DME bits are outside of the specified range [(43)], optionally including tunable tolerances (margins) and hysteresis.
Note that impulse noise of high energy may saturate the receivers, making them sample values of the analog signal that are seemingly within specified range or virtually zero-valued, however the time series of such an event often have distinct characteristics, as follows:

- While a fully saturated differential receiver may see a signal that is within the allowed range of $1\,\mathrm{V}\,(0.5\,\mathrm{V_{RMS}})\pm 20\%$ defined by the standard [(29)], the sampled values are very close to zero in case of saturation, and a maintained signal level of near zero is not valid for DME decoding;
- Due to the capacitance of the Media Dependent Interface (MDI), the saturation of the receiver does not happen instantaneously, therefore – depending on the underlying architecture – PMA may be able to reliably assume saturation even in these cases.

These, and possibly some additional mechanisms, allow the PMA_RX to convey not only DME bit and 5B symbol estimators, but also side-channel information regarding the validity of each of these, this way enabling erasure detection and correction as well.

4.6.5.C Configurable burst length via interleaving

With the channel model introduced in Section 4.2 the well-known technique of **interleaving** can be used to improve the burst error correction capabilities of the code according to known results inside and outside of the domain of Ethernet [77, 87]. When used with interleaving of depth L, the RSC codeword of length n gets combined with $L-1$ other interleaved RSC codewords to form a **superblock** of length nL 5B symbols.

Such technique allows achieving the per-frame configurable burst error resilience discussed in Section 1.1 and Section 4.5.2, as the overall burst error correcting capability of a scheme that builds on an interleaving of depth L is tL, which means that the 6 DME bit times of disturbing effect per stimulus pulse mentioned in Section 4.2 may be eliminated using a very simple 2-interleaver, which is capable of dealing with any error in the DME domain of $[6, 10]$, or erasures of $[16, 20]$ bits in length.

4.6.6 An example $\{19, 19\}$ coding scheme

In this section details and methods related to constructing an actual $\{19, 19\}$ coding scheme are presented. The details of the content here are at a level that is sufficient to be used directly by an implementation. The section does not recommend an actual implementation, as the construct does not restrict any known implementation technique to be effective.

4.6.6.A Selection of free parameters

Section 4.6 showed the theoretical constructs behind a $\{c, u\}$ coding scheme, while in this section we present the full details of an actual working example coding scheme based on that. This example will cover the processes of the encoding and that of the decoding under the following set of constant parameters:

- *Finite field*: GF(32);
- *Field generator polynomial*: $p(x) = x^5 + x^2 + 1$;
- $t = 1$;
- *Type of the underlying ECC*: shortened $(19, 17)$ systematic RSC;
- σ *for the code generator polynomial of* $g(x) = (x + \alpha^{\sigma})(x + \alpha^{\sigma+1})$: $\sigma = 0$;
- $c = 19$;
- $u = 19$;
- *Codeword layout*: as per Figure 4.6;
- *Value for* FECESD: $\mathtt{X} = 0$ (binary 00000);
- *Bit position* where FSs among **DS**s *match*: Least Significant Bit (LSB).

From these the following can be derived:

- All elements of GF(32) are shown in Table 4.6
- *Code generator polynomial*: $g(x) = x^2 + 3x + 2$, thus $\lambda = 3$, and $\omega = 2$;
- *Number of complete* **DS**s: 15;
- *Number of* **SB**s: 9;
- *Number of* **PS**s: 2;
- $\mathsf{P1}_{\text{left}} = 15a + 7b$;
- $\mathsf{P2}_{\text{left}} = 14a + 6b$;
- $S_1(0) = \{0, 2, 4, 6, 8, 10, 12, 14, 17, 19, 21, 23, 25, 27, 29\}$;
- $S_1(1) = \{1, 3, 5, 9, 11, 15, 16, 18, 20, 22, 24, 26, 28, 30\}$;
- $S_2(0) = \{0 - 6, 8 - 12, 14 - 15\}$;
- $S_2(1) = \{16 - 30\}$;
- $S_1(0) \cap S_2(0) = \{0, 2, 4, 6, 8, 10, 12, 14\}$,
  $S_1(1) \cap S_2(0) = \{1, 3, 5, 9, 11, 15\}$,
  $S_1(0) \cap S_2(1) = \{16, 18, 20, 22, 24, 26, 28, 30\}$, and
  $S_1(1) \cap S_2(1) = \{17, 19, 21, 23, 25, 27, 29\}$,
  thus $s_{\text{sig}} = 6$

This $\{19, 19\}$ coding scheme builds on $d = \lfloor 4u/5 \rfloor = 15$ complete **DS**s, which – according to Equation (4.2) – needs $G(15) = 54$ FTRs. This is achieved by breaking down each FTR index (denoted by FTR_{idx}) into a high part (denoted by FTR_{high}, encoded by $\mathsf{S5}$–$\mathsf{S9}$) and a low part (denoted by FTR_{low}, encoded by $\mathsf{S1}$–$\mathsf{S4}$) using any mapping that is **injective** and the **inverse** of which **is surjective**. One such example is shown by Equation (4.15).

$$\mathrm{FTR}_{\mathrm{idx}} = 9(\mathrm{FTR}_{\mathrm{high}} - 1) + \mathrm{FTR}_{\mathrm{low}} \tag{4.15}$$

Now we have everything to lay out the lookup tables that map the admissible symbols as required. In this example we do this using the most natural – incremental – approach, as follows:

- Table 4.8: The 28 values used for transcoding FSs among complete **DS**s are assigned to (f, δ) pairs such, that it follows the order of appearance of f in Table 4.3, while δ increases sequentially;
- Table 4.7: The 54 FTRs are assigned indices in increasing order of FTR position and complexity;
- Table 4.9: The $G(15)/s_{\mathrm{sig}} = 9$ values for $\mathrm{FTR}_{\mathrm{low}}$ are assigned in increasing order;
- Table 4.10: Finally, the $s_{\mathrm{sig}} = 6$ values for $\mathrm{FTR}_{\mathrm{high}}$ are assigned in increasing order as well, while maintaining the presence of values from $S_1(0)$, $S_1(1)$, $S_2(0)$, and $S_2(1)$ so that Equation (4.5) remains satisfied all the way.

  For example, in the group where $\mathrm{FTR}_{\mathrm{high}} = 1$ the following applies:
  - $2 \in S_1(0)$ and $2 \in S_2(0)$;
  - $17 \in S_1(0)$ and $17 \in S_2(1)$;
  - $1 \in S_1(1)$ and $1 \in S_2(0)$;
  - $16 \in S_1(1)$ and $16 \in S_2(1)$.

Table 4.6: Elements of GF(32) with the field generator polynomial $p(x) = x^5 + x^2 + 1$

Index and polynomial forms	Decimal (binary) values	Index and polynomial forms	Decimal (binary) values
0	0 (00000)	$\alpha^{15} = \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$	31 (11111)
$\alpha^0 = 1$	1 (00001)	$\alpha^{16} = \alpha^4 + \alpha^3 + \alpha + 1$	27 (11011)
$\alpha^1 = \alpha$	2 (00010)	$\alpha^{17} = \alpha^4 + \alpha + 1$	19 (10011)
$\alpha^2 = \alpha^2$	4 (00100)	$\alpha^{18} = \alpha + 1$	3 (00011)
$\alpha^3 = \alpha^3$	8 (01000)	$\alpha^{19} = \alpha^2 + \alpha$	6 (00110)
$\alpha^4 = \alpha^4$	16 (10000)	$\alpha^{20} = \alpha^3 + \alpha^2$	12 (01100)
$\alpha^5 = \alpha^2 + 1$	5 (00101)	$\alpha^{21} = \alpha^4 + \alpha^3$	24 (11000)
$\alpha^6 = \alpha^3 + \alpha$	10 (01010)	$\alpha^{22} = \alpha^4 + \alpha^2 + 1$	21 (10101)
$\alpha^7 = \alpha^4 + \alpha^2$	20 (10100)	$\alpha^{23} = \alpha^3 + \alpha^2 + \alpha + 1$	15 (01111)
$\alpha^8 = \alpha^3 + \alpha^2 + 1$	13 (01101)	$\alpha^{24} = \alpha^4 + \alpha^3 + \alpha^2 + \alpha$	30 (11110)
$\alpha^9 = \alpha^4 + \alpha^3 + \alpha$	26 (11010)	$\alpha^{25} = \alpha^4 + \alpha^3 + 1$	25 (11001)
$\alpha^{10} = \alpha^4 + 1$	17 (10001)	$\alpha^{26} = \alpha^4 + \alpha^2 + \alpha + 1$	23 (10111)
$\alpha^{11} = \alpha^2 + \alpha + 1$	7 (00111)	$\alpha^{27} = \alpha^3 + \alpha + 1$	11 (01011)
$\alpha^{12} = \alpha^3 + \alpha^2 + \alpha$	14 (01110)	$\alpha^{28} = \alpha^4 + \alpha^2 + \alpha$	22 (10110)
$\alpha^{13} = \alpha^4 + \alpha^3 + \alpha^2$	28 (11100)	$\alpha^{29} = \alpha^3 + 1$	9 (01001)
$\alpha^{14} = \alpha^4 + \alpha^3 + \alpha^2 + 1$	29 (11101)	$\alpha^{30} = \alpha^4 + \alpha$	18 (10010)

Table 4.7: An example assignment of FTR indices and their meanings in a $\{19, 19\}$ coding scheme

$\mathrm{FTR}_{\mathrm{idx}}$	$\mathrm{FTR}_{\mathrm{high}}$	$\mathrm{FTR}_{\mathrm{low}}$	A	B	$\mathrm{FTR}_{\mathrm{idx}}$	$\mathrm{FTR}_{\mathrm{high}}$	$\mathrm{FTR}_{\mathrm{low}}$	A	B
1	1	1	–		..				
2	1	2	D1	–	**37**	5	1	D3	δ of 6th FS $+= 1{\cdot}7$
3	1	3	D2		**38**	5	2	D4	δ of 1st FS $+= 1{\cdot}7$
..					..				
9	1	9	D8	–	**42**	5	6	D4	δ of 5th FS $+= 1{\cdot}7$
10	2	1	D9		**43**	5	7	D5	δ of 1st FS $+= 1{\cdot}7$
16	2	7	D15		..				
17	2	8	D1	δ of 1st FS $+= 1{\cdot}7$	**46**	6	1	D5	δ of 4th FS $+= 1{\cdot}7$
18	2	9	D1	δ of 2nd FS $+= 1{\cdot}7$	**47**	6	2	D6	δ of 1st FS $+= 1{\cdot}7$
..					..				
24	3	6	D1	δ of 8th FS $+= 1{\cdot}7$	**49**	6	4	D6	δ of 3rd FS $+= 1{\cdot}7$
25	3	7	D2	δ of 1st FS $+= 1{\cdot}7$	**50**	6	5	D7	δ of 1st FS $+= 1{\cdot}7$
..					**51**	6	6	D7	δ of 2nd FS $+= 1{\cdot}7$
28	4	1	D2	δ of 4th FS $+= 1{\cdot}7$	**52**	6	7	D8	δ of 1st FS $+= 1{\cdot}7$
..					**53**	6	8	D1	δs of 1st and 2nd FS $+= 1{\cdot}7$
31	4	4	D2	δ of 7th FS $+= 1{\cdot}7$	**54**	6	9	D1	δ of 1st FS $+= 2{\cdot}7$
32	4	5	D3	δ of 1st FS $+= 1{\cdot}7$					

Legend:
A: The first FS among the **DS**s
B: Special Constellation Descriptor (SCD)
Each (A, B) pair uniquely identifies an FTR

Example in which an FTR has previously been used	Applicable $\mathrm{FTR}_{\mathrm{idx}}$
Example #1 in Figure 4.7 on page 60	**1**
Example #2 in Figure 4.7 on page 60	**2**
Example #3 in Figure 4.7 on page 60	**16**
Example #4 in Figure 4.7 on page 60	**10**
Example #5 in Figure 4.8 on page 61	**10**
Example #6 in Figure 4.8 on page 61	**18**
Example #7 in Figure 4.8 on page 61	**25**
Example #8 in Figure 4.8 on page 61	**54**
Example #9 in Figure 4.8 on page 61	**53**
In Figure 4.13 on page 75	**28**

Table 4.8: An example assignment of 5B symbol values and transcoding functions usable by the **DS** of any $\{c, u\}$ coding scheme, if `X` is zero (binary 00000)

Binary, **decimal** values	f	δ	Binary, **decimal** values	f	δ	Binary, **decimal** values	f	δ	Binary, **decimal** values	f	δ
00001, **1**	`T`	EoL	01010, **10**	`T`	2	10011, **19**	`T`	4	11011, **27**	`T`	6
00010, **2**	`R`		01011, **11**	`R`		10100, **20**	`R`		11100, **28**	`R`	
00011, **3**	`I`		01100, **12**	`I`		10101, **21**	`I`		11101, **29**	`I`	
00100, **4**	`X`		01110, **14**	`X`		10110, **22**	`X`		11110, **30**	`X`	
00101, **5**	`T`	1	01111, **15**	`T`	3	10111, **23**	`T`	5			
00110, **6**	`R`		10000, **16**	`R`		11000, **24**	`R`				
01000, **8**	`I`		10001, **17**	`I`		11001, **25**	`I`				
01001, **9**	`X`		10010, **18**	`X`		11010, **26**	`X`				

Table 4.9: An example assignment of 5B symbol values and FTR_{low} usable by S1–S4 of a $\{19, 19\}$ coding scheme

Binary, **decimal** values	FTR_{low}	Binary, **decimal** values	FTR_{low}	Binary, **decimal** values	FTR_{low}
0001, **1**	1	0101, **5**	4	1001, **9**	7
0010, **2**	2	0111, **7**	5	1010, **10**	8
0100, **4**	3	1000, **8**	6	1011, **11**	9

Table 4.10: An example assignment of 5B symbol values and FTR_{high} usable by S5–S9 of a $\{19, 19\}$ coding scheme

Binary, **decimal** values	FTR_{high}	Binary, **decimal** values	FTR_{high}	Binary, **decimal** values	FTR_{high}
00001, **1**	1	00101, **5**	3	01010, **10**	5
00010, **2**		00110, **6**		01011, **11**	
10000, **16**		10100, **20**		11000, **24**	
10001, **17**		10101, **21**		11001, **25**	
00011, **3**	2	01000, **8**	4	01100, **12**	6
00100, **4**		01001, **9**		01111, **15**	
10010, **18**		10110, **22**		11010, **26**	
10011, **19**		10111, **23**		11011, **27**	

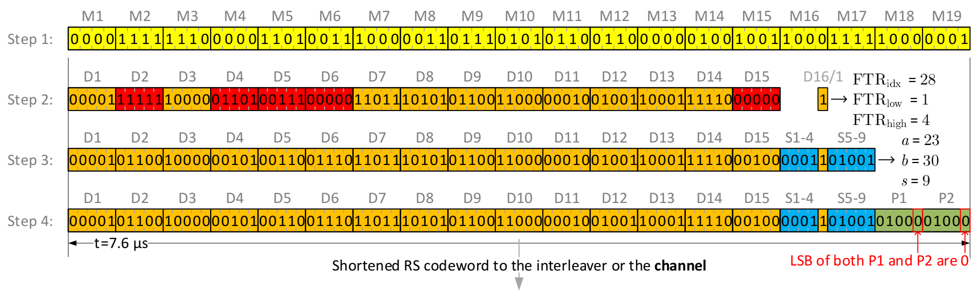


Figure 4.13: A complete encoding process in an example {19, 19} coding scheme

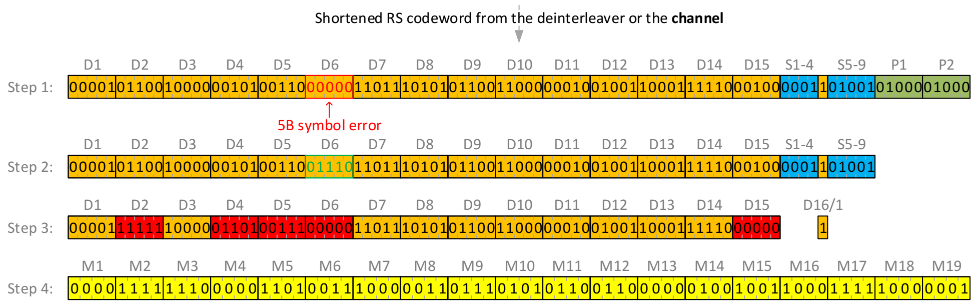


Figure 4.14: A complete decoding process in an example {19, 19} coding scheme

4.6.6.B The encoding process

The example encoding shown in Figure 4.13 [60] consists of the following main steps:

1. $u = 19$ user data 4B nibbles (**MS**s) from the MAC/PLS interface are conveyed through the MII and collected by the encoder;
2. The **MS**s are scrambled;
3. Those are arranged into 5B symbols (**DS**s) according to the selected layout (in this case according to Figure 4.6), FSs are identified, which determines the values for FTR_{idx}, providing FTR_{high} and FTR_{low} through the selected mapping, which in this case is as in Equation (4.15);
4. All FSs are transcoded using the linked-list, S1–S4 are filled-in based on FTR_{low}, then a and b are calculated, which – in conjunction with FTR_{high} – are used to determine s, to be used directly as S5–S9 in the codeword;
5. The result of this process is fed to the (19, 17) systematic RSC encoder of choice, which provides the codeword with $2t = 2$ **PS**s that is free of FSs.

The output codeword of this process can either be forwarded directly to the channel, or to an L-interleaver that produces a **superblock**, which can then be conveyed to the channel.

4.6.6.C The decoding process

The example decoding shown in Figure 4.14 (60) is just the reversal (inversion) of the encoding process shown in Figure 4.13, and it goes as follows:

1. The codeword **estimator**, is received from the channel or from the L-deinterleaver, optionally with side-channel information on erasures, from the PMA_RX;

2. The codeword is fed to the RSC decoder. If the codeword has neither errors nor erasures it passes the codeword along, otherwise it may operate one of the following ways:

   - In case of $t = 1$ 5B symbol error **or** not more than $2t = 2$ 5B symbol erasures, these are guaranteed to be corrected in all cases;
   - In case of errors and erasures that go above these limits, it does one of the following 2 things:
     - It signals non-correctable errors/erasures: Such signal may be used to provide higher overall error resilience;
     - Or it may do a miss-correction: In this case it is ultimately the responsibility of the Frame Check Sequence (FCS) *and* the higher layer protocols (if any) to detect frame corruption.

3. FTR_{idx} is decoded and the linked-list is walked, to invert the FS transcoding applied by the encoder;

4. The user data is separated into 4B nibbles, considering the framing introduced in Section 4.6.2;

5. The **MS**s are descrambled and get conveyed to the MAC/PLS interface via the MII, clocked on a per-symbol basis.

This process is guaranteed to be able to correct any burst errors consisting of any combination of up to L consecutive 5B symbols.

4.7 Comparison with other solutions

The work presented here falls into the category of "error-correcting constrained coding" within line coding. This area is generally well-established [88] and is being actively researched. Some work [89] presents a single-error-correcting method that utilizes *markers* and can deal with both synchronization (bit-slip) and additive errors as well, using a technique that is applicable to DC-free codes. The use of such construct is limited however by the low rates it offers, as well as the fact that synchronization does not pose problems in case of 10BASE-T1S, due to DME modulation it anyway uses.

Some other work [90] proposes schemes that are not based on 5B symbols or the multiples of these, therefore those do not meet the requirements posed by interleaving explained in Section 4.5.2.

RLL ECCs are well known and can offer a rate that asymptotically approaches the capacity of the (d, k) constraint [91]. Very recent research [92] also aims at implementing capacity-approaching constrained codes that can deal with single *substitution*. Other recent work [93] established the so-called "RLL-ECC" scheme on top of Hamming and LDPC codes, that is applicable only when the control matrix manifests certain properties. These however specifically target RLL constraints, while the Forbidden

Symbols (FSs) pose a different challenge and RLL codes cannot be used to deal with FSs.

We believe that our work opens a new sub-category within constrained error correcting codes by presenting methods that are capable of dealing with FSs in cases where existing techniques are not applicable. Nevertheless, it is worth comparing the $\{c, u\}$ coding scheme with other, possibly applicable approaches, and such analysis is to be provided in this section.

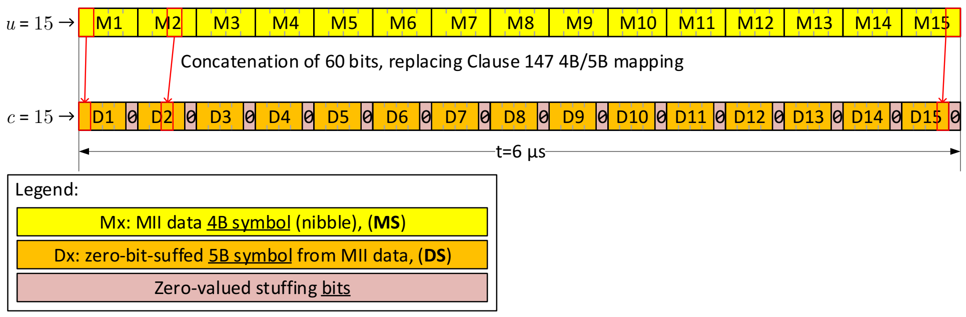

$u = 15 \rightarrow$
M1
M2
M3
M4
M5
M6
M7
M8
M9
M10
M11
M12
M13
M14
M15
Concatenation of 60 bits, replacing Clause 147 4B/5B mapping
$c = 15 \rightarrow$
D1 0 D2 0 D3 0 D4 0 D5 0 D6 0 D7 0 D8 0 D9 0 D10 0 D11 0 D12 0 D13 0 D14 0 D15 0
t=6 μs
Legend:
Mx: MII data 4B symbol (nibble), (MS)
Dx: zero-bit-suffed 5B symbol from MII data, (DS)
Zero-valued stuffing bits

Figure 4.15: A possible bit stuffing process attempting to deal with 15 **MS**s

A naïve approach will be discussed in details in Section 4.8.1, therefore we spare no further explanation on that here.

On the other hand, bit stuffing and other bit insertion techniques, such as padding are worth the analysis. These techniques insert *non-information* bits into the blocks or stream of bits on the output of the encoder to meet certain criteria. In case of 10BASE-T1S, bit stuffing is applicable to the problem at hand for example by following the observations made in Section 4.6.3.B, so that a zero bit is inserted after each **MS**, this way guaranteeing that that no **DS** would end up being an FS. One reason why this may not lead to a complete solution is the rate-preserving requirement set in Section 4.3.2.B: if the remaining (unused) redundancy present in the 4B/5B mapping defined by Clause 147, shown in Table 4.2, is used up completely by the bit stuffing, then no space remains in the codeword based on which parity bits/symbols could be formed, while preserving the rate, as shown by Figure 4.15 in comparison with Figure 4.5. One may argue that stuffing could be applied only when FS happens to appear, but in that case signaling capability is required to indicate where stuffing was used, which leads to a contradiction.
To restate the problem and to extend the explanation given in Section 4.1.2, it can be seen that the difficulty of the problem lies in finding the appropriate scheme to implement the following 3 features in conjunction, while maintaining low complexity:

- Provide free bits for the ECC to be implementable;
- Avoid formation of FSs by user data;
- Make sure that the encoding process itself does not create new FSs, for example at the parity bits/symbols of the codeword.

While RLL codes are not applicable to the problem for similar reasons, it is worth noting that the DME modulation used by 10BASE-T1S is a.k.a. Frequency Modulation (FM), which is a half-rate RLL that is being considered as a means to further extend the error and/or erasure correcting capabilities of the $\{c, u\}$ coding scheme (see Section 5.2).

4.8 Evaluation of performance

There are multiple metrics along which evaluation of performance may be done, however most of these – for example encoding/decoding time, memory demand, silicon gate count – are implementation-specific and are difficult to find useful general bounds on. Two metrics however can be meaningfully evaluated without assuming a specific implementation: these are the **efficiency** of the coding scheme and the **coding delays**.

4.8.1 Efficiency of the code

It is well worth evaluating how economically the $\{c, u\}$ coding scheme fills the code-space created by the free admissible symbols in the codeword. This may be done using Equation (4.16), which expresses the number of **bits** that are unused by each existing $\{c, u\}$ coding scheme listed in Table 4.5.

$$E(c,u) = \log_2\left(\frac{(|\mathrm{GF}(32)| - 3\cdot|\mathcal{F}_\mathrm{b}|)\cdot|\mathcal{A}_\mathrm{p}|^d\cdot|\mathcal{A}_\mathrm{b}|^{c-d-3}\cdot G(d)}{16^u\cdot H(\ell)}\right) \tag{4.16}$$

Obviously, implementing an actual scheme where $E(c, u) = 0$ is not feasible even in systems where the implementation is less constrained. This is true at least due to the following 2 reasons:

- As hinted in Section 4.6.3.B, such system would need a lookup table that has $32\cdot32 = 1024$ elements, to be able to pick s perfectly, depending on the actual values of a and b. If however such solution was implemented, it would increase the signaling capability of s from $s_\mathrm{sig} = 7$ to $(|\mathrm{GF}(32)| - 3\cdot|\mathcal{F}_\mathrm{b}|) = 23$;
- Additionally, such solution would require capability of quickly dividing, calculating the modulo of, and multiply numbers that are in $[0, 16^u]$.

The efficiency of all existing $\{c, u\}$ coding schemes is summarized in Table 4.11, from which it is visible that the $\{c, u\}$ – most notably the $\{19, 19\}$ – coding scheme is highly efficient, while remaining low-complexity from the perspective of implementation.

Table 4.11: Efficiency of all existing $\{c, u\}$ coding schemes

$\{c,u\}$	$\lfloor E(c,u)\rfloor$	$\{c,u\}$	$\lfloor E(c,u)\rfloor$
$\{19,19\}$	4	$\{28,28\}$	5
$\{20,20\}$		$\{25,26\}$	6
$\{21,21\}$	5	$\{26,26\}$	
$\{22,22\}$		$\{28,29\}$	
$\{23,23\}$		$\{29,29\}$	
$\{24,24\}$		$\{29,30\}$	
$\{24,25\}$		$\{30,30\}$	
$\{25,25\}$		$\{30,31\}$	
$\{26,27\}$		$\{31,31\}$	
$\{27,27\}$		$\{31,32\}$	
$\{27,28\}$		$\{31,33\}$	

4.8.2 Coding delays

In general, systematic error correcting codes have the advantage of being encodable “on-the-fly”, which means that when the data arrives to the encoder, it can immediately start forming the codewords and producing output without – or with insignificant – delay. However in case of the proposed $\{c, u\}$ coding scheme, this does not stand. The reason for this is three-fold:

- The user input data concatenation process shown in Figure 4.5 causes a delay on the output stream. As an example to be able to create D12, M15 must be received by the encoder;
- The escaped elements of the linked-list and the values of **SB**s may be formed only after the last **MS** has been received from the MII;
- When interleaving is applied, it imposes an additional fixed delay, as the superblock can be formed only after L RSC codewords become available.

An upper **bound** on the total delay caused by these factors is $5cL$ bit times, as $5cL$ expresses the size of the superblock. As an example, without interleaving ($L = 1$), the encoding delay of a $\{19, 19\}$ coding scheme is $\approx 7.6\,\mu s$, while with minimum interleaving ($L = 2$) it is $\approx 15.2\,\mu s$.

In contrast with this, a PHY that has no FEC in it must rely on **retransmissions**, the lower bound for the delay of which is the total transmission time for the packet. In case of Ethernet this is 512 bit times (20), in addition to the delays imposed by the higher layers coordinating the retransmissions. The packet retransmission delays are – typically several magnitudes – larger than those attributed to the proposed $\{c, u\}$ coding scheme with any reasonable value for L, irrespective of the details of the higher layers' mechanisms that trigger the retransmissions (such as whether positive or a negative acknowledgment schemes are used there).

Additionally, architectures that hide or incorporate the MII interface [20, 22, 24] may partially or completely eliminate encoding and decoding delay.

4.9 Conclusions on Forward Error Correction

In this chapter, we have established the foundations for the need for an FEC for 10BASE-T1M that is backward-compatible with 10BASE-T1S. Through a systematic approach, we have presented the difficulties associated with solving this problem and have brought forward new coding methods and their unique combination to form a complete solutions.

Through formal methods and the analysis of these, we have proposed a general construct for a novel, systematic RSC-based $\{c, u\}$ coding scheme, that can correct burst errors and – optionally – burst erasures as well, in a way that these FEC-based Ethernet frames are guaranteed not to disrupt the operations 10BASE-T1S nodes or the PLCA cycle of the mixing segment. Next, we have discussed limits of the scheme and how to alleviate for those, then, we have shown the construction method and all the details of an actual $\{19, 19\}$ coding scheme that may be used by itself, or in conjunction with interleaving to achieve per-frame configurable burst error and erasure resilience. Finally the performance of the proposed scheme has been evaluated with respect to efficiency and coding delays, then it has been compared to other possible solutions. We believe that this theme of our work opens a new sub-category within constrained error correcting codes.

Chapter 5 Closing thoughts

5.1 Summary and conclusions

In the dissertation we have aimed at providing a set of enhancements to the recently released 10 Megabit(s) per Second (Mb/s) single-pair Ethernet Physical Layer Device (PHY) called 10BASE-T1S. Our research journey has led us to being able to propose a **new set of frame preambles**, as well as observations regarding the validity of earlier research results published during IEEE Project 802.3cg [4] (P802.3cg).

Shortly after the conclusion of the research on the preamble, the new IEEE Project 802.3da [29] (P802.3da) has been formed and approved to operate. In one hand, the consensually defined objectives of P802.3da have demanded the enhanced version of 10BASE-T1S – referred to as 10BASE-T1M – to remain interoperable with Clause 147, made the application of the newly found optimal preambles difficult in P802.3da. On the other hand however, these preambles are still available to be used beyond P802.3da, as a replacement for the current preamble, as well as a parallel preamble to implement additional functionality, such as Operations, Administration, and Maintenance (OAM) and Out of Band (OoB) signaling.

As research continued under the umbrella of P802.3da, but with unchanged research focus of improving noise immunity of 10BASE-T1M, we targeted adding **per-frame configurable burst error and erasure correcting capability** to 10BASE-T1M in a way that is **backward-compatible with 10BASE-T1S**. Through the definition of, and carefully combined application of 3 new techniques, we proposed a coding scheme that is capable of achieving this goal even in the presence of low-complexity requirements for the new PHY.

Both themes of the research have been guided, validated and tested through the software implementations of a correlator, an analog channel model, a 10BASE-T1S PHY, including all of its Finite State Machines (FSMs), and the entirety of the proposed $\{c, u\}$ coding scheme for 10BASE-T1M. These efforts also included the implementation and exhaustive use of a test and validation bench in the digital domain.

5.2 Future work

The $\{c, u\}$ coding scheme has been presented at P802.3da [44, 45], where it received overwhelmingly positive acceptance and requests for additional information [94]. To continue research efforts meaningfully however, the decision in the project on whether the interoperability objective (27) is to stay in the project must be resolved. The P802.3da Task Force (TF) is actively researching and discussing this at the time of writing the dissertation. If the objective remains in the project, then the following improvements to the Forward Error Correction (FEC) proposed here would be the most natural next steps:

- Establishing an additional method that provides burst error resilience to the preamble as well. As the $\{c, u\}$ coding scheme already addressed the resilient signaling of the end-of-frame (ESD), covering the preamble would make the extended solution usable “as is”;
- The $\{c, u\}$ coding scheme looks at the channel as a single binary stream of bits, however, Clause 147 defines Differential Manchester Encoding (DME) modulation as line code. As DME has a simple, strict set of rules, both error and

erasure detection performance may be improved based on these. Understanding the applicability and the extent of this potential of improvement may be a worthy future direction.

If however the interoperability objective (27) is abolished, it opens different possibilities, namely in the direction of proposing an FEC that is not backward-compatible with Clause 147, but better supports reach extension, possibly over a rate-preserving non-binary modulation.

Additionally, it might be interesting and useful to look at the extension of the Galois Field beyond GF(32). There are other commonly used legacy encodings, such as the 8B/10B mapping used in a variety of gigabit communication, and the 64B/66B [95] family used in 2.5 Gb/s and higher rate Ethernet, which might benefit from the proposed scheme.

Acknowledgments

Ph.D. candidate wishes to express his sincere appreciation to a handful of experts in various roles, whose contribution and support allowed this research to take place and to conclude.

Endless gratitude goes to **Hiroyoshi Morita** and **Satoshi Ohzahata**. Professor Morita and Professor Ohzahata, who – as the academic advisors of the candidate – allowed overcoming major challenges and difficulties along this long journey of research and development that led to the results presented in the dissertation and beyond.

George Zimmerman has been one of the key figures not only for a large set of Ethernet PHYs, including 10BASE-T1S and 10BASE-T1L, but his whole career is the best role models for how to solve difficult technological and management problems, without losing sight of what is important. Dr. Zimmerman has made a large set of contributions to the development of the $\{c, u\}$ coding scheme, allowing it to be one of the key feature candidates for 10BASE-T1M.

Piergiorgio Beruto's deep knowledge and enlightening insights helped to overcome difficulties and avoid dead-ends along the way. Mr. Beruto and his team are also the "fathers" (original inventors) of Physical Layer Collision Avoidance (PLCA), this way making a remarkable contribution to the body of knowledge and technology available today, and for the generations to come to build on.

Last, but not least, **Ari Kattainen** has been strong supporter of various activities that allowed P802.3cg to conclude the way it did. Mr. Kattainen has provided his personal trust, endless patience and support to the candidate, which allowed him to become and operate as an editor for the Institute of Electrical and Electronics Engineers (IEEE) standard 802.3cg-2019, which permitted him to make contributions to the efforts of the TF, while also learning how bleeding edge technology development goes in front and behind the curtain.

It has been a long and lasting pleasure for the candidate to research under the guidance of, and to work with the experts acknowledged here and beyond.

List of Symbols and Notations

All indices start counting from one, in other words the indices are 1-based.
Fractional values are rounded to two decimal places by default.
Relevant nomenclature from IEEE is often used.

General

$\lvert\mathcal{C}\rvert$	Cardinality of set $\mathcal{C}$
(i, j)	$\{x \in \mathbb{N} \mid i < x < j\}$
$[i, j]$	$\{x \in \mathbb{N} \mid i \leq x \leq j\}$

In Chapter 3

τ	Section 3.2.2 on page 29
$\mathrm{ACC}_{s_a,s_b}(\tau)$	Equation (3.3) on page 28
$\mathrm{AAC}_{s_a}(\tau)$	Equation (3.4) on page 29
$\mathrm{AAC}_{s}(\tau)$	Equation (3.5) on page 29
$\mathrm{AAC}_{s,n}$	Section 3.3.3.A on page 31
$\mathrm{G}_{\mathrm{AAC}}$	Equation (3.6) on page 33
$\mathrm{G}_{\mathrm{ACC}}$	Equation (3.7) on page 33
L	Section 3.2.2 on page 29
$\ell 1$	Section 3.3.3.A on page 31
$\ell 2-$	Section 3.3.3.A on page 31
$\ell 2+$	Section 3.3.3.A on page 31
$\ell 1_n$	Section 3.3.3.A on page 31
$\ell 2-_n$	Section 3.3.3.A on page 31
$\ell 2+_n$	Section 3.3.3.A on page 31
s_1	Section 3.2.2.A on page 29
s_2	Section 3.4.1 on page 35
s_3	Section 3.4.1 on page 35
s_6	Section 3.2.2.A on page 29
$s_{6,n}$	Section 3.3.3.A on page 31
s_a	Section 3.2.2 on page 28
s_b	Section 3.2.2 on page 28

In Chapter 4

δ	Section 4.6.3.A on page 59
δ_{act}	Section 4.6.3.A on page 59
$\delta_{\max}$	Section 4.6.3.A on page 59
λ	Section 4.6.3.B on page 62
σ	Section 4.6.3.B on page 62
ω	Section 4.6.3.B on page 62
$\mathcal{A}_{\mathrm{b}}$	Section 4.6.3.B on page 65
$\mathcal{A}_{\mathrm{p}}$	Section 4.6.3 on page 65
$E(c, u)$	Equation (4.16) on page 78
$\mathcal{F}_{\mathrm{b}}$	Section 4.6.2 on page 65
$\mathcal{F}_{\mathrm{p}}$	Section 4.3.5 on page 65
$\mathrm{FTR}_{\mathrm{idx}}$	Equation (4.15) on page 72
$\mathrm{FTR}_{\mathrm{high}}$	Equation (4.15) on page 72
$\mathrm{FTR}_{\mathrm{low}}$	Equation (4.15) on page 72
$G(d)$	Equation (4.2) on page 62
$H(\ell)$	Equation (4.11) on page 67
L	Section 4.6.5.C on page 70
$\mathrm{LSB\_of}(n)$	Section 4.6.3.B on page 65
$S_1(v)$	Equation (4.7) on page 65
$S_2(v)$	Equation (4.6) on page 65
a	Section 4.6.3.B on page 62
b	Section 4.6.3.B on page 62
c	Section 4.6.1 on page 55
$\{c, u\}$	Section 4.6.1 on page 55
d	Section 4.6.1 on page 56
$d_{\min}$	Section 4.5 on page 52
$e(j)$	Section 4.6.3.A on page 61
f	Section 4.6.3.A on page 59
$g(x)$	Section 4.6.3.B on page 62
$h(i)$	Equation (4.10) on page 67
k	Section 4.5 on page 52
ℓ	Section 4.6.3.C on page 66
m	Section 4.6.3.A on page 60
n	Section 4.5 on page 52
$p(x)$	Section 4.6.3.C on page 66
r	Section 4.6.4 on page 67
s_{sig}	Equation (4.9) on page 66
t	Section 4.5 on page 52
u	Section 4.6.1 on page 55
x	Section 4.6.3.A on page 59
y	Section 4.6.3.A on page 59
z	Section 4.6.3.A on page 60

List of Terms and Abbreviations

μC	Microcontroller
μP	Microprocessor
10BASE-T1L	10 Mb/s Baseband Single-Pair Ethernet PHY for long reach
10BASE-T1S	10 Mb/s Baseband Single-Pair Ethernet PHY for short reach
10BASE-T1x	10BASE-T1S and/or 10BASE-T1L
10SPE	10 Mb/s Single-Pair Ethernet
AAC	Aperiodic Autocorrelation
ACC	Aperiodic Cross Correlation
ADC	Analog-to-Digital Converter
AFE	Analog Front-End
AN	Auto-Negotiation
ANBCWN	Added Narrow-Band Continuous Wave Noise
AWGN	Additive White Gaussian Noise
BCH	Bose-Chaudhuri-Hocquenghem
BEACON	The 5B symbol sequence of `NNNNN`, with a time-duration of $2\,\mu s$
BER	Bit Error Rate
BoF	Beginning of Frame
BPF	Band-Pass Filter
CAN	Controller Area Network
CIR	Carrier-to-Interference Ratio
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
DADR	Destination Address, sometimes also abbreviated as “DA”
DADR	Source Address, sometimes also abbreviated as “SA”
DFE	Decision Feedback Equalization
DFFT	Discrete Fast Fourier Transform
DLL	Data Link Layer
DME	Differential Manchester Encoding
DS	5B “Data Symbol” (see Section 4.6.1)
ECC	Error Correcting Code
EEE	Energy Efficient Ethernet
EFT	Electrical Fast Transient
EMI	Electromagnetic Interference
EoL	End of List
EPON	Ethernet Passive Optical Network
ESD	End Sequence Delimiter
EtherCAT	Ethernet for Control Automation Technology
FCS	Frame Check Sequence
FD	Full Duplex
FEC	Forward Error Correction
FECESD	Noise-resilient ESD used by the proposed FEC scheme
FM	Frequency Modulation
FS	Forbidden Symbol
FSM	Finite State Machine
FTR	FS Transcoding Recipe
Gb/s	Gigabit(s) per Second
GCS	Golay Complementary Sequence
GCSP	GCS-based Preamble
GF	Galois Field

Gs/s	Gigasample(s) per Second
HD	Half Duplex
HDD	Harness Defect Detection
$\mathbf{I^2C}$	Inter-Integrated Circuit
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IL	Insertion Loss
IoT	Internet of Things
ISI	Intersymbol Interference
ISO	International Organization for Standardization
IT	Information Technology
LAN	Local Area Network
LCM	Lifecycle Management
LDPC	Low-Density Parity-Check
LFSR	Linear-Feedback Shift Register
LON	Local Operating Network
LPF	Low-Pass Filter
LSB	Least Significant Bit
M2M	Machine-to-Machine
MAC	Media Access Control
MAC-PHY	See Section 2.3.3
MAC/PLS	Media Access Control/Physical Signaling
MAN	Metropolitan Area Network
Mb/s	Megabit(s) per Second
MCM	Max Cable Model
MDI	Media Dependent Interface
MDROP	Multidrop
MDS	Maximum Distance Separable
MII	Media Independent Interface
MS	4B "MII Symbol" (see Section 4.6.1)
Ms/s	Megasample(s) per Second
NRZ	Non-Return-to-Zero
OA	OPEN Alliance
OAM	Operations, Administration, and Maintenance
ODVA	Open Devicenet Vendors Association
OoB	Out of Band
OSI	Open Systems Interconnection
OT	Operational Technology
P802.3cg	IEEE Project 802.3cg [4]
P802.3da	IEEE Project 802.3da [29]
P&P	Plug-and-Play
PAR	Project Authorization Request
PCB	Printed Circuit Board
PCM	Product Change Management
PCS	Physical Coding Sublayer
PCS_RX	Physical Coding Sublayer's Receive Function
PCS_TX	Physical Coding Sublayer's Transmit Function
PD	Powered Device
PHY	Physical Layer Device
PLCA	Physical Layer Collision Avoidance

PLCA-ID	PLCA Identifier
PLCA_CTRL	PLCA Control Function
PLCA_DATA	PLCA Data Function
PLS	Physical Signaling
PMA	Physical Medium Attachment Sublayer
PMA_RX	Physical Medium Attachment Sublayer's Receive Function
PMA_TX	Physical Medium Attachment Sublayer's Transmit Function
PMD	Physical Medium Dependent Sublayer
PoE	Power over Ethernet
PS	5B "Parity Symbol" (see Section 4.6.1)
PSD	Power Spectral Density
PSE	Power Sourcing Equipment
Pt2Pt	Point-to-Point
RevCom	IEEE Standards Review Committee
RL	Return Loss
RLL	Run-length Limited
RNG	Random Number Generator
RS	Reconciliation Sublayer
RSC	Reed-Solomon Error and Erasure Correcting Code
SA	Standards Association
SASB	IEEE SA SB
SB	A (single) "Signaling Bit" (see Section 4.6.1)
SCD	Special Constellation Descriptor
SCM	Source Control Management
SFD	Start of Frame Delimiter
SIG	Special Interest Group
SNR	Signal-to-Noise Ratio
SPC	Single Parity Check
SPI	Serial Peripheral Interface
SPMD	Single-Pair Multidrop
SQI	Signal Quality Indicator
SR	Sample Rate
Tb/s	Terabit(s) per Second
TC14	Technical Committee 14
TC6	Technical Committee 6
TDMA	Time-Division Multi(ple) Access
TF	Task Force
TO	Transmit Opportunity
TP	Twisted Pair
TPMR	Two-port MAC Relay
TSN	Time-Sensitive Networking
TSSI	Time Synchronization Service Interface
USB	Universal Serial Bus
VLAN	Virtual LAN
VPN	Virtual Private Networking
WAN	Wide Area Network
WG	Working Group
XOR	Exclusive Or

List of Equations

List of Tables

List of Figures

List of References

[1] Andrew Tanenbaum and David Wetherall. *Computer Networks (5th edition).* Pearson, 2010. ISBN: 9780132126953.

[2] Institute of Electrical and Electronics Engineers (IEEE) Standards Association (SA). *IEEE 802.3-2018 - IEEE Standard for Ethernet.* Aug. 31, 2018. URL: `https://standards.ieee.org/standard/802_3-2018.html`.

[3] Jim Eggers and Steve Hodnett. *Ethernet Autonegotiation Best Practices.* Sun BluePrints OnLine. Sun Microsystems. July 2004. URL: `https://www.netcraftsmen.com/wp-content/uploads/2019/04/Ethernet-Autonegotiation-Best-Practices.pdf`.

[4] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3cg. *IEEE P802.3cg 10 Mb/s Single Pair Ethernet Task Force.* URL: `https://www.ieee802.org/3/cg/`.

[5] Institute of Electrical and Electronics Engineers (IEEE) 802. *IEEE 802 Overview.* June 15, 2015. URL: `https://www.ieee802.org/1/files/public/docs2015/admin-newcomer-orientation-for-CPRI-0615.pdf`.

[6] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3cg. *Objectives for 802.3cg.* Mar. 18, 2018. URL: `https://www.ieee802.org/3/cg/objectives_3cg_0318.pdf`.

[7] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3cg. *Drafts for 802.3cg.* URL: `https://www.ieee802.org/3/cg/private/index.html`.

[8] George Zimmerman et al. *IEEE P802.3cg 10 Mb/s Single Pair Ethernet: A guide.* Jan. 16, 2019. URL: `https://www.ieee802.org/3/cg/public/Jan2019/Tutorial_cg_0119_final.pdf`.

[9] Institute of Electrical and Electronics Engineers (IEEE) Standards Association (SA). *IEEE 802.3cg-2019 - IEEE Standard for Ethernet - Amendment 5: Physical Layer Specifications and Management Parameters for 10 Mb/s Operation and Associated Power Delivery over a Single Balanced Pair of Conductors.* Nov. 7, 2019. URL: `https://standards.ieee.org/standard/802_3cg-2019.html`.

[10] David D. Brandt, Chirag Malkan, Tony Wang, and Jeff Martin. “Enhancements to Single-pair Ethernet for Constrained Devices”. In: *ODVA 2020 Industry Conference.* Mar. 4, 2020. URL: `https://www.odva.org/wp-content/uploads/2020/05/2020-ODVA-Conference_SPE_Constrained_Malkan_Wang_Brandt_Martin_Final.pdf`.

[11] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3cg. *Root Project Authorization Request (PAR) for 802.3cg.* Dec. 7, 2016. URL: `https://www.ieee802.org/3/cg/IEEE_P802_3cg_PAR_071216.pdf`.

[12] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3cg. *Revised (final) Project Authorization Request (PAR) for 802.3cg.* May 14, 2018. URL: https://www.ieee802.org/3/cg/P802_3cg_PAR_140518.pdf.

[13] Ari Kattainen and Gergely Huszak. *IEEE 802.3cg 10 Mb/s Single Twisted Pair Ethernet Elevator/Escalator Use Case, Topology and Failure Modes (rev1).* URL: https://www.ieee802.org/3/cg/public/Sept2017/huszak_3cg_01a_0917.pdf and https://www.ieee802.org/3/cg/public/Sept2017/Huszak_3cg_02a_0917.pdf. Sept. 13, 2017.

[14] Mats Björkman and Bob Melanerd. "Impact of the Ethernet Capture Effect on Bandwidth Measurements". In: *NETWORKING 2000*. Vol. LNCS 1815. 2020, pp. 156–167. DOI: 10.1007/3-540-45551-5_14. URL: https://link.springer.com/content/pdf/10.1007/3-540-45551-5_14.pdf.

[15] Luc van Dijk and Günter Sporer. "Functional Safety for Automotive Ethernet Networks". In: *Journal of Traffic and Transportation Engineering* 6.4 (Aug. 2018). DOI: 10.17265/2328-2142/2018.04.003. URL: https://www.davidpublisher.com/Public/uploads/Contribute/5b88ec8f4b032.pdf.

[16] Piergiorgio Beruto and Antonio Orzelli. *Proposal for short-reach multi-drop 10M SPE (former PLCA).* Sept. 14, 2018. URL: https://www.ieee802.org/3/cg/public/Sept2017/Beruto_3cg_01a_0917.pdf.

[17] Piergiorgio Beruto and Antonio Orzelli. *802.3cg draft 2.0 PLCA (Clause 148) Overview.* URL: https://www.ieee802.org/3/cg/public/July2018/PLCAoverview.pdf and https://www.ieee802.org/3/cg/public/July2018/PLCAFAQ.pdf. July 9, 2018.

[18] Sujan Pandey, Dr. Philip Axer, and Don Pannell. *PLCA Data-Rate Fairness.* Mar. 2018. URL: https://www.ieee802.org/3/cg/public/Mar2018/Pandey_3cg_01a_0318.pdf.

[19] ChangYoung Jo, JaeWan Park, and JaeWook Jeon. "PLCA Experiment for 10Base-T1 Performance Verification". In: *2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia).* Nov. 2020. DOI: 10.1109/ICCE-Asia49877.2020.9277440. URL: https://ieeexplore.ieee.org/document/9277440.

[20] CanovaTech. *CT25207: Simplified Multidrop 10BASE-T1S Ethernet PHY with MAC Controller for Intra-System application.* URL: https://www.canovatech.com/ct25207.html.

[21] ChangYoung Jo, JaeWan Park, and JaeWook Jeon. *10BASE-T1x MAC-PHY Serial Interface.* URL: http://www.opensig.org/download/document/263/OPEN_Alliance_10BASET1x_MACPHY_Serial_Interface_V1.00.pdf.

[22] CanovaTech.
CT25206: Multidrop 10BASE-T1S Ethernet PHY with MAC Controller.
URL: `https://www.canovatech.com/ct25206.html`.

[23] OPEN Alliance TC14. *10BASE-T1S Transceiver Interface (v1.4).* July 22, 2020.
URL: `https://members.opensig.org/wg/TC14/document/6553`.

[24] Analog Devices. *ADIN1110: Preliminary Data Sheet.*
URL: `https://www.analog.com/media/en/technical-documentation/data-sheets/adin1110.pdf`.

[25] Bosch. *CAN XL – The Next Step in CANn Evolution.* Feb. 2021.
URL: `https://www.bosch-semiconductors.com/media/ip_modules/pdf_2/can_xl_1/canxl_intro_20210225.pdf`.

[26] Dong Gun Kam, Heeseok Lee, Jonghoon Kim, and Joungho Kim.
"A New Twisted Differential Line Structure on High-Speed Printed Circuit Boards to Enhance Immunity to Crosstalk and External Noise". In: *Microwave and Wireless Components Letters (IEEE)* 13 (Oct. 2003), pp. 411–413.
DOI: `10.1109/LMWC.2003.815181`.
URL: `https://ieeexplore.ieee.org/abstract/document/1232563`.

[27] International Electrotechnical Commission (IEC).
IEC 61000-4-4:2012, Electromagnetic compatibility (EMC) - Part 4-4: Testing and measurement techniques - Electrical fast transient/burst immunity test.
Apr. 30, 2012. URL: `https://webstore.iec.ch/publication/4222`.

[28] George Zimmerman.
Enhanced Multidrop with Interoperability – what do we need? July 20, 2020.
URL: `https://www.ieee802.org/3/da/public/jul20/zimmerman_3da_071020.pdf`.

[29] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3da. *IEEE P802.3da 10 Mb/s Single Pair Multidrop Segments Enhancement Task Force.*
URL: `https://www.ieee802.org/3/da/`.

[30] Institute of Electrical and Electronics Engineers (IEEE) 802 LAN/MAN Standards Committee.
Root Project Authorization Request (PAR) for 802.3da. June 2, 2020.
URL: `https://www.ieee802.org/3/da/P802d3da_PAR.pdf`.

[31] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3da. *Objectives for 802.3da.*
URL: `https://www.ieee802.org/3/da/802d3da_objectives.pdf`.

[32] Gergely Huszak. *IEEE 802.3cg 10 Mb/s Single Twisted Pair Ethernet Elevator/Escalator Use Case.* Aug. 30, 2017. URL: `https://www.ieee802.org/3/cg/public/adhoc/802%203cg%20Use%20Case,%20final.pdf`.

[33] Ari Kattainen and Gergely Huszak. *IEEE 802.3cg 10 Mb/s Single Twisted Pair Ethernet Elevator/Escalator Use Case, Topology and Failure Modes (rev6).*
Nov. 7, 2017. URL: `https://www.ieee802.org/3/cg/public/Nov2017/kattainen_huszak_3cg_01b_1117.pdf`.

[34] Institute of Electrical and Electronics Engineers (IEEE) 802.
Comment entry tutorial. URL: `https://www.ieee802.org/3/ballots/p802d3_comment_entry_tutorial_v1p1.pdf`.

[35] Gergely Huszak and Hiroyoshi Morita.
"On the 10BASE-T1S preamble for multidrop".
In: *International Global Information Infrastructure Symposium (GIIS).*
Dec. 2019, pp. 1–6. DOI: 10.1109/GIIS48668.2019.9044963.
URL: https://ieeexplore.ieee.org/document/9044963.

[36] Tim Baggett, Piergiorgio Beruto, Gergely Huszak, and David Law.
Resolution to r02-33 and PLCA state diagram, including figures. Aug. 2019.
URL: https://www.ieee802.org/3/cg/public/Aug2019/r02-33%20Proposed%20Response.pdf.

[37] Piergiorgio Beruto Gergely Huszak George Zimmerman.
802.3cg: On the PCS Receive in clause 147. May 2019.
URL: https://www.ieee802.org/3/cg/public/May2019/Huszak_Zimmerman_Beruto_cg_01_01a_T1S_PCS_RX.pdf.

[38] Piergiorgio Beruto and Gergely Huszak. *PLCA burst mode.* Sept. 14, 2018.
URL: https://www.ieee802.org/3/cg/public/Sept2018/beruto_huszak_plca_bursting.pdf.

[39] Chris DiMinico, Bob Voss, and Paul Wachtel.
SPE Multidrop Enhancements Mixing Segment Considerations Update.
July 20, 2020. URL: https://www.ieee802.org/3/da/public/jul20/diminico_SPMD_01_0720.pdf.

[40] M.E.Austin. "Decision-feedback equalization for digital communication over dispersive channels". In: 1967.
URL: https://dspace.mit.edu/bitstream/handle/1721.1/4282/RLE-TR-461-04744914.pdf.

[41] Sergey Miropolsky and Stephan Frei.
"Comparability of RF Immunity Test Methods for IC Design Purposes".
In: *8th Workshop on Electromagnetic Compatibility of Integrated Circuits* (2011), pp. 59–64. URL: http://www.bordsysteme.tu-dortmund.de/publications/2011_EMC_Compo_Dubrovnik_Comparability_of_RF_Immunity_Test_Methods_for_IC_Design_Purposes.pdf.

[42] Jason Potterf. *SPMD Lessons Learned from PoE.* May 6, 2020.
URL: https://www.ieee802.org/3/SPMD/public/may0620/SPMD_Lessons_Learned_From_PoE.pdf.

[43] Gergely Huszak, Hiroyoshi Morita, and George Zimmerman.
"Backward-compatible forward error correction of burst errors and erasures for 10BASE-T1S". In: *IEICE Transcations on Communications* E104-B.12 (Dec. 2021), pp. 1524–1538. ISSN: 1745-1345. DOI: 10.1587/transcom.2021EBP3016.
URL: https://search.ieice.org/bin/summary.php?id=e104-b_12_1524&category=B&year=2021&lang=E&abst=.

[44] Gergely Huszak and George Zimmerman. *FEC for 802.3da.* June 16, 2021.
URL: https://www.ieee802.org/3/da/public/061621/huszak_zimmerman_fec_3da_06162021.pdf.

[45] Gergely Huszak and George Zimmerman.
Technical and implementation details on the FEC for 802.3da. July 14, 2021.
URL: https://www.ieee802.org/3/da/public/0721/huszak_zimmerman_fec_3da_07142021.pdf.

[46] Gergely Huszak. *Verbatim 10BASE-T1S PHY simulator.* URL: `https://github.com/ghuszak/public/tree/master/simu/`.

[47] Gergely Huszak. *Verification of the proposed backward-compatible forward error correcting scheme for 10BASE-T1S.* URL: `https://github.com/ghuszak/public/tree/master/FEC_data`.

[48] Jason Potterf. *802.3da Clause 147 Multidrop Interoperability.* Sept. 8, 2021. URL: `https://www.ieee802.org/3/da/public/090821/SPMD_Potterf_Clause_147_T1S_Backwards_Compatibility_2021-09-08.pdf`.

[49] Gergely Huszak. *On the meaning of "interoperability" in objective #4 of 802.3da.* Oct. 6, 2021. URL: `https://www.ieee802.org/3/da/public/100621/huszak_da_01_100621.pdf`.

[50] David D. Brandt. *IEEE P802.3da Interoperability Objective Clarification.* Oct. 6, 2021. URL: `https://www.ieee802.org/3/da/public/100621/brandt_3da_01_092121.pdf`.

[51] Gergely Huszak. *Mixing Segment Recovery/Redundancy: Termination.* July 14, 2020. URL: `https://www.ieee802.org/3/da/public/jul20/huszak_3da_071020.pdf`.

[52] David D. Brandt. *10BASE-T1S Increased Transmit Voltage.* Nov. 7, 2018. URL: `https://www.ieee802.org/3/cg/public/adhoc/brandt_3cg_01_1118_stg_comments.pdf`.

[53] David D. Brandt. *10BASE-T1S Optional Increased Transmit Voltage.* Nov. 15, 2018. URL: `https://www.ieee802.org/3/cg/public/Nov2018/brandt_3cg_01e_1118.pdf`.

[54] Sumeeth Nagaraj, Sheehan Khan, Christian Schlegel, and Marat Burnashev. "On Preamble Detection in Packet-Based Wireless Networks". In: *IEEE International Symposium on Spread Spectrum Techniques and Applications.* Oct. 2006, pp. 476–480. DOI: `10.1109/ISSSTA.2006.311817`. URL: `https://ieeexplore.ieee.org/document/4100606`.

[55] José Herrera-Bustamante, Vanessa Rodríguez-Ludeña, A.G. Correa-Mena, and Diego Barragán-Guerrero. "Design and implementation in USRP of a preamble-based synchronizer for OFDM systems". In: *2020 IEEE ANDESCON.* 2020, pp. 1–6. DOI: `10.1109/ANDESCON50619.2020.9272183`. URL: `https://ieeexplore.ieee.org/document/9272183`.

[56] Jung-Hyun Park, Min-Ho Kim, Doo-Chan Hwang, and Jungil Han. "Analysis on Preamble Detection and False Alarm Probabilities of ICPC Method". In: *2015 IEEE 82nd Vehicular Technology Conference (VTC2015-Fall).* 2015, pp. 1–5. DOI: `10.1109/VTCFall.2015.7390844`. URL: `https://ieeexplore.ieee.org/document/7390844`.

[57] P.G.A. Jespers, M.G. Windal, and T.G. Watteyne. "An Integrated Binary Correlator Module". In: *IEEE Journal of Solid-State Circuits* 18.3 (June 1983), pp. 286–290. DOI: `10.1109/JSSC.1983.1051941`. URL: `https://ieeexplore.ieee.org/document/1051941`.

[58] Marcel J. E. Golay. "Multi-Slit Spectrometry". In: *Journal of the Optical Society of America* 39.6 (1949), pp. 437–444. DOI: `10.1364/JOSA.39.000437`.

[59] Gian Marco Bo and Piergiorgio Beruto. *MDI electrical specification.* May 2018. URL: https://www.ieee802.org/3/cg/public/May2018/beruto_3cg_02_0518.pdf.

[60] Piergiorgio Beruto and Antonio Orzelli. *Collision Detection Reliability in 10BASE-T1S.* Mar. 6, 2019. URL: https://www.ieee802.org/3/cg/public/adhoc/beruto_3cg_collision_detection.pdf.

[61] Wai Ho Mow and S.-Y.R. Li. "Aperiodic Autocorrelation and Crosscorrelation of Polyphase Sequences". In: *IEEE Transactions on Information Theory* 43.3 (May 1997), pp. 1000–1007. DOI: 10.1109/18.568711. URL: https://ieeexplore.ieee.org/document/568711.

[62] Piergiorgio Beruto and Antonio Orzelli. *Short Reach PCS, PMA and PLCA baseline proposal.* Nov. 7, 2017. URL: http://www.ieee802.org/3/cg/public/Nov2017/beruto_3cg_01a_117.pdf.

[63] Jay Cordaro, Ahmad Chini, and Mehmet Tazebay. *New Preamble Proposal for 10BASE-T1S.* Dec. 20, 2017. URL: https://www.ieee802.org/3/cg/public/adhoc/cordaro_8023cg_short_reach_new_preamble_proposal_1220.pdf.

[64] Jay Cordaro. *Proposed Preamble: Synchronization and Harness Defect Detection.* Apr. 11, 2018. URL: http://www.ieee802.org/3/cg/public/adhoc/cordaro_3cg_06_0418.pdf.

[65] Piergiorgio Beruto and Antonio Orzelli. *T1S preamble.* May 22, 2018. URL: https://www.ieee802.org/3/cg/public/May2018/beruto_3cg_03_0518.pdf.

[66] George C. Clark Jr. and J. Bibb Cain. *Error-Correction Coding for Digital Communications.* Springer, May 27, 2013. ISBN: 0306406152.

[67] Working Party XV/4. *Comparison Bose-Chaudhuri-Hocquenghem BCH and Reed Solomon (Doc. #476).* Comité Consultatif International Téléphonique et Télégraphique (CCITT) SGXV. Specialists Group on Coding for Visual Telephony. URL: https://www.itu.int/wftp3/av-arch/video-site/h261/H261_Specialists_Group/Contributions/476.pdf.

[68] Jay Cordaro and Mehmet Tazebay. *OAM for 802.3cg 10BASE-T1S.* May 2018. URL: https://www.ieee802.org/3/cg/public/May2018/8023cg10base-t1s%20OAM.PDF.

[69] Cameron Jones. *Noise Immunity for Single Pair Cabling and Applicability to Industrial Applications.* July 17, 2020. URL: https://grouper.ieee.org/groups/802/3/da/public/jul20/Noise_Considerations_SPE.pdf.

[70] C. E. Shannon. "A mathematical theory of communication". In: *The Bell System Technical Journal* 27.3 (July 1948), pp. 379–423. DOI: 10.1002/j.1538-7305.1948.tb01338.x. URL: https://ieeexplore.ieee.org/document/6773024.

[71] Wojciech Koczwara and George Zimmerman. *Impulse Immunity and FEC Objective.* URL: https://www.ieee802.org/3/SPMD/public/jan20/Koczwara_Zimmerman_3SPMD_01_0120.pdf.

[72] George Zimmerman. *Control of FEC in Multidrop: Introducing the MSL Client.* July 14, 2021. URL: https://www.ieee802.org/3/da/public/0721/zimmerman_3da_01_07142021.pdf.

[73] Kees Schouhamer Immink. *Codes for Mass Data Storage Systems.* Shannon Foundation Publishers, 2004. ISBN: 9074249272.

[74] Gi Lee Byeong and Chang Kim Seok. *Scrambling Techniques for Digital Transmission.* Springer Verlag, 1994. ISBN: 9780387198637.

[75] Piergiorgio Beruto and Antonio Orzelli. *PLCA Burst mode.* Oct. 24, 2018. URL: https://www.ieee802.org/3/cg/public/adhoc/beruto_3cg_PLCA_burst_mode_revA%20.pdf.

[76] W. Cary Huffman and Vera Pless. *Fundamentals of Error-Correcting Codes (1st edition).* Cambridge University Press, 2010. ISBN: 9780521131704.

[77] Ron Roth. *Introduction to Coding Theory.* Cambridge University Press, 2006. ISBN: 9780521845045.

[78] Institute of Electrical and Electronics Engineers (IEEE) Standards Association (SA). *IEEE 802.3ch-2020 - IEEE Standard for Ethernet - Amendment 8: Physical Layer Specifications and Management Parameters for 2.5 Gb/s, 5 Gb/s, and 10 Gb/s Automotive Electrical Ethernet.* June 30, 2020. URL: https://ieeexplore.ieee.org/document/9146430.

[79] Cathy Liu. "100+ Gb/s Ethernet Forward Error Correction (FEC) Analysis". In: *Signal Integrity Journal* (2019). URL: https://www.signalintegrityjournal.com/articles/1286-gbs-ethernet-forward-error-correction-fec-analysis?v=preview.

[80] L.R. Vermani. *Elements of Algebraic Coding Theory.* Chapman and Hall/CRC, 1996. ISBN: 9780412573804.

[81] R.C. Bose and D.K. Ray-Chaudhuri. "On a class of error correcting binary group codes". In: *Information and Control* 3.1 (1960), pp. 68–79. DOI: 10.1016/S0019-9958(60)90287-4. URL: https://www.sciencedirect.com/science/article/pii/S0019995860902874.

[82] Alexis Hocquenghem. "Codes correcteurs d'erreurs". In: *Chiffres* 2 (1952).

[83] L. J. Deutsch. "The Effects of Reed-Solomon Code Shortening on the Performance of Coded Telemetry Systems". In: *The Telecommunication and Data Acquisition Progress Report* (Nov. 1983), pp. 14–20. URL: https://ipnpr.jpl.nasa.gov/progress_report/42-75/75B.PDF.

[84] A.J. Han Vinck. *Coding Concepts and Reed-Solomon Codes.* Institute For Experimental Mathematics, Essen, Germany. University of Duisburg-Essen, Germany. 2013. URL: https://www.uni-due.de/imperia/md/images/dc/book_coding_concepts_and_reed_solomon_codes.pdf.

[85] I. Reed and G. Solomon. "Polynomial Codes Over Certain Finite Fields". In: *Journal of The Society for Industrial and Applied Mathematics* 8.2 (1960), pp. 300–304. DOI: 10.1137/0108018. URL: https://faculty.math.illinois.edu/~duursma/CT/RS-1960.pdf.

[86] C.K.P. Clarke. *Reed-Solomon error correction.* BBC Research & Development. British Broadcasting Corporation. Jan. 2002. URL: https://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP031.pdf.

[87] Pete Anslow. *RS(544,514) FEC performance with 4:1 interleaving.* Sept. 6, 2018. URL: https://www.ieee802.org/3/ck/public/18_09/anslow_3ck_01_0918.pdf.

[88] An Introduction to Coding for Constrained Systems. *Brian H. Marcus and Ron M. Roth and Paul H. Siegel (5th edition).* Oct. 2001. URL: https://ronny.cswp.cs.technion.ac.il/wp-content/uploads/sites/54/2016/05/chapters1-9.pdf.

[89] WA Clarke, Albert Helberg, and Hendrik Christoffel Ferreira. "Constrained codes for the correction of synchronization and additive errors". In: *IEEE South African Symposium on Communications and Signal Processing.* Sept. 1993, pp. 58–62. ISBN: 0780312929. DOI: 10.1109/COMSIG.1993.365873.

[90] S. Manoj and C. Babu. "Improved error detection and correction for memory reliability against multiple cell upsets using DMC & PMC". In: *IEEE Annual India Conference (INDICON) 2016.* 2016, pp. 1–6. DOI: 10.1109/INDICON.2016.7839094.

[91] M. Siala and G.K. Kaleh. "Joint multilevel RLL and error correction coding". In: 1995, pp. 201–. DOI: 10.1109/ISIT.1995.531875.

[92] Thanh Nguyen, Kui Cai, Kees Schouhamer Immink, and Han Mao Kiah. "Capacity-Approaching Constrained Codes with Error Correction for DNA-Based Data Storage". In: vol. 67. Aug. 2021, pp. 5602–5613. DOI: 10.1109/TIT.2021.3066430.

[93] Peter Farkaš and Frank Schindler. "Run length limited error control codes construction based on one control matrix property". In: vol. 68. Jan. 2017, pp. 322–324. DOI: 10.1515/jee-2017-0046.

[94] Institute of Electrical and Electronics Engineers (IEEE) Task Force (TF) 802.3da. *Minutes of IEEE 802.3da SPMD TF meeting (July Plenary 2021).* July 14, 2021. URL: https://www.ieee802.org/3/da/public/0721/spmd_TF_plenary_minutes_00_0721.pdf.

[95] Marek Hajduczenia. *64b/66b line code.* Mar. 19, 2013. URL: https://www.ieee802.org/3/bn/public/mar13/hajduczenia_3bn_04_0313.pdf.

[96] SciPy.org. *Python-based ecosystem of open-source software for mathematics.* URL: https://www.scipy.org/.

[97] SciPy.org. *SciPy library: Fundamental library for scientific computing.* URL: https://www.scipy.org/scipylib/index.html.

[98] NumPy.org. *NumPy library: The fundamental package for scientific computing with Python.* URL: https://numpy.org/.

[99] MatPlotLib.org. *Visualization with Python.* URL: https://matplotlib.org/.

Appendix

A.1 Simulations related to Chapter 3

Simulations supporting research on the preamble consisted of the following two efforts, both implemented in Python, relying on various components of the **SciPy ecosystem** [96]:

- First, the complete channel model described in Section 3.2.1 has been established. This part of the simulations are described subsequently in Section A.1.1;
- Next, the channel model has been applied to generate and analyze inputs and output of the simulated channel. This phase of the simulations are explained in Section A.1.2.

A.1.1 Channel model

Figure 3.2 shows 7 components comprising the channel simulator. Each of these has been implemented using the following components of the SciPy ecosystem:

- Establishing digital Butterworth Low-Pass Filters (LPFs) and Band-Pass Filters (BPFs) using SciPy libraries [97] for elements #2 and #6 of the channel model shown in Figure 3.2: See Algorithm 3 on page 97;
- Application of signal filters also for the same 2 elements of the channel model: See Algorithm 4 on page 98;
- Inducing maximum losses using Discrete Fast Fourier Transforms (DFFTs) for the Max Cable Model (MCM), which is element #4 of the channel model: See Algorithm 5 on page 98;
- **NumPy** [98]: Various random number generation and array management – including trigonometric – functions to handle Added Narrow-Band Continuous Wave Noise (ANBCWN) for elements #3 and #5 of the channel model: See Algorithm 1 on page 97 and Algorithm 6 on page 98;

A.1.2 Correlators

Correlators and the analysis of their output have been done using the following components of the SciPy ecosystem:

- **Cross-correlating** digital representation of analog signals according to Equation (3.3): See Algorithm 2 on page 97;
- **Visualization** (2D plotting) [99]: 2-dimensional display of processed waveforms created during the research, for example:
  - `matplotlib.pyplot.scatter();`
  - `matplotlib.pyplot.annotate();`
  - `matplotlib.pyplot.plot();`
  - `matplotlib.pyplot.show().`

A.1.3 Details of algorithms

Python algorithms shown here are the core elements of simulations presented in Section A.1. The purpose of showing these details is to allow the implementation of new simulations that produce comparable results.

Algorithm 1 Generation of ANBCWN in Python

```
# Generate all the ANBCWNs
for freq in (numpy.arange(1, (30 + 0.5), 0.5) * 1000000):
  for phase in numpy.arange(0, (2 * math.pi), (math.pi / 4)):
    count = (((2 * math.pi) * freq) / (SAMPLE_RATE / len_axis_x_preamble))
    xs = numpy.linspace(start = phase,
                        stop = (phase + count),
                        num = len_axis_x_preamble,
                        endpoint = True)
    anbcwn = numpy.sin(xs)
    anbcwns.append(anbcwns / 2)
```

Algorithm 2 Application of AAC in Python

```
# Aperiodic Autocorrelation (AAC), according to Equation 3.5
aac = numpy.core.multiarray.correlate(a = analog_samples_s1,
                                      v = analog_samples_s6,
                                      mode = 2)
```

Algorithm 3 Definition for the Butterworth filters in Python

```
# 2nd order digital Butterworth low-pass filter (LPF)
# with a corner frequency of CORNER_FREQ:
fbl, fal = scipy.signal.butter(Wn = (CORNER_FREQ / (SAMPLE_RATE / 2)),
                               N = 2,
                               btype = 'low',
                               analog = False)
# 2nd order digital Butterworth high-pass filter (HPF)
# with a corner frequency of CORNER_FREQ:
fbh, fah = scipy.signal.butter(Wn = (CORNER_FREQ / (SAMPLE_RATE / 2)),
                               N = 2,
                               btype = 'high',
                               analog = False)
```

Algorithm 4 Application of digital filters in Python

```
# Apply a low-pass filter (LPF) to input signal 'signal_in',
# producing filtered output signal 'signal_out':
signal_out = scipy.signal.lfilter(b = fbl, a = fal, x = signal_in)
# Apply a low-pass filter (HPF) to input signal 'signal_in',
# producing filtered output signal 'signal_out':
signal_out = scipy.signal.lfilter(b = fbh, a = fah, x = signal_in)
# Apply a band-pass filter (BPF) to input signal 'signal_in',
# producing filtered output signal 'signal_out':
signal_tmp = scipy.signal.lfilter(b = fbh, a = fah, x = signal_in)
signal_out = scipy.signal.lfilter(b = fbl, a = fal, x = signal_tmp)
```

Algorithm 5 Application of the filters

```
# Compute Discrete Fourier Transform for analog samples
sx_zs = scipy.fftpack.rfft(x = sx_analog_samples)
# Get the Discrete Fourier Transform sample frequencies
sx_dfft_frequencies = scipy.fftpack.rfftfreq(n = sx_analog_samples,
                                             d = (1 / SAMPLE_RATE))
# Compute inverse Discrete Fourier Transform
sxplus1_analog_samples = scipy.fftpack.irfft(x = sx_zs)
```

Algorithm 6 Generation of AWGN in Python

```
# Generate AWGN
numpy.random.normal(loc = 0,
                    scale = (math.pow(10, (-30 / 20)) / math.sqrt(2 / m.pi)),
                    size = len_axis_x_preamble)
```

B.2 Simulations related to Chapter 4

The development process of the FEC has been tightly coupled with the implementation of 3 independent software projects, as discussed in subsequent sections.

B.2.1 Implementation of 10BASE-T1S

To be able to test observations fundamental to the entirety of the research (including the proposed FEC scheme), at the very beginning of the research a complete implementation of IEEE Std 802.3cg-2019 [9] Clause 147 and Clause 148 has been made [46]. The code is verbatim to the standard in the sense that it implements all the following functionality, exactly as written in the text:

- All FSMs of the following functions:
  - Physical Medium Attachment Sublayer's Receive Function (PMA_RX);
  - Physical Medium Attachment Sublayer's Transmit Function (PMA_TX);
  - Physical Coding Sublayer's Receive Function (PCS_RX);
  - Physical Coding Sublayer's Transmit Function (PCS_TX);
  - PLCA Control Function (PLCA_CTRL);
  - PLCA Data Function (PLCA_DATA).
- The scrambler and descrambler as Linear-Feedback Shift Registers (LFSRs);
- The Media Independent Interface (MII).

To allow the Physical Layer Device (PHY) to operate, a general timer module and a simple Media Access Control/Physical Signaling (MAC/PLS) have also been added. The channel has been modeled as a shared digital pipe, where collisions were handled exactly as written in the standard (67).

The number of stations connected to the simulated mixing segment is configurable and the simulator runs each station – the coordinator, as well as the followers – under the same set of rules.

The implementation [46] followed the very development of the standard in Source Control Management (SCM), and it was used also to detect errors in some versions of the draft, thus directly contributing to the improvement of the standard [37, 36].

Observations and conclusions made for example in Section 4.3 have all been based on, and have been verified by this simulator.

The simulator realization is in C language, consisting of over 4000 lines of source code.

B.2.2 Implementation of a $\{19, 19\}$ coding scheme

This implementation [47] consists of two phases:

1. The earlier implements the complete example $\{19, 19\}$ coding scheme as presented in Section 4.6.6;
2. The latter is to test the correctness of the complete encoding and decoding process, as explained in Section 4.6.6.B and Section 4.6.6.C (respectively), with a single 5B symbol error, and single and double 5B symbol erasures.

(67) IEEE Std 802.3cg-2019 [9] "147.3.5 Collision detection".

The exhaustive test verifying the implementability, applicability, and correctness of the scheme's error correcting capabilities follows the algorithm shown in Section B.2.3 on page 100. While that algorithm is represented as pseudo-code, however the actual simulation system developed as part of the research was written in C language, without the use of any external libraries, and it consists of over 2000 lines of source code. Erasure correcting capabilities of the code have been verified through similar methods. Additionally, the methods explained throughout Section 4.6.3.A have also been verified, through the following programmatic mechanism:

1. Generate a possible codeword of the $\{c, u\}$ coding scheme being analyzed;
2. Produce the FS Transcoding Recipe (FTR) for the codeword;
3. Check whether that FS Transcoding Recipe (FTR) has already been recognized earlier:
    - If the FTR is new, add it to the set of necessary FTRs;
    - Otherwise do nothing, as FTR has already been found.
4. Using the same process, iterate through all possible locations and values of the FSs in the $\{c, u\}$ coding scheme being analyzed;
5. At the end, count the recognized FTRs collected during all iterations.

The output of this exhaustive process has confirmed the correctness of the content of Section 4.6.3.A, including that of Equation (4.2), for all necessary values of d.

B.2.3 Details of algorithms

```
function GET_RANDOM_ADMISSIBLE_5B_SYM
    ▷ Generate a single 5B symbol that is not 0 (X), 7 (R), 13 (R), or 31 (I)
    return a single 5B symbol
end function

function GET_RANDOM_BIT
    ▷ Generate a random bit (0 or 1)
    return a single bit
end function

function GET_RANDOM_ERROR
    ▷ Generate a random, non-zero 5B symbol used as a symbol error
    return a single 5B symbol
end function

function PLACE_DSS($DSs\_1\_to\_15$, $bit\_1\_of\_D16$)
    ▷ Create the DS part of the codeword by placing 76 bits (MSs) as shown in
      Figure 4.6
    return the DS part of the codeword
end function
```

```
function ENCODE_SCHEME($DSs$)
    ▷ Implement steps 2 and 3 of Figure 4.13 in Section 4.6.6.B
    return the DS and SB part of the codeword
end function

function ENCODE_RSC($DSs\_and\_SBs$)
    ▷ Implement step 4 of Figure 4.13 in Section 4.6.6.B
    return the fully encoded RSC codeword
end function

function COUNT_FSs($codeword$)
    ▷ Checks if the codeword includes any FSs
    return True or False
end function

function DECODE_SCHEME($codeword$)
    ▷ Implement steps 2 and 3 of Figure 4.14 in Section 4.6.6.B
    return the 76 user bits (MSs)
end function

procedure EXIT($result$)
    ▷ Exit program and return final results as per $result$
      – If $result$ = True, return with SUCCESS
      – If $result \neq$ True, return with FAILURE
end procedure

function GENERATE_76_DS_BITS(s)
    ▷ Generate 19 MSs, total of 76 bits
      $s$: an integer in $[0, 5^{15} - 1]$ representing D1-D15 and it works as follows:
      1. Consider $s$ to be an integer with radix 5, having 15 characters
      2. Get the character at the $i^{\text{th}}$ place of $s$ and do as follows:
      – If that character is 0, then make the $i^{\text{th}}$ DS an X (0)
      – If that character is 1, then make the $i^{\text{th}}$ DS an R (7)
      – If that character is 2, then make the $i^{\text{th}}$ DS an T (13)
      – If that character is 3, then make the $i^{\text{th}}$ DS an I (31)
      – Otherwise make the $i^{\text{th}}$ DS a random admissible 5B symbol
    ▷ Output: $retval\_DSs$ an array of 15 5B symbols, holding D1-D15 as
      shown in Figure 4.6
    for $i \leftarrow 0$ to 15 do
        if $i = 0$ then
            $\_s \leftarrow s$
        else
            $\_s \leftarrow \lfloor s/5^i \rfloor$
        end if

        $\_m \leftarrow \_s \bmod 5$
        if $\_m = 0$ then                                    ▷ Generate X (0)
            $retval\_DSs[i] \leftarrow 0$
        else if $\_m = 1$ then                               ▷ Generate R (7)
            $retval\_DSs[i] \leftarrow 7$
        else if $\_m = 2$ then                               ▷ Generate T (13)
            $retval\_DSs[i] \leftarrow 13$
        else if $\_m = 3$ then                               ▷ Generate I (31)
            $retval\_DSs[i] \leftarrow 31$
        else                                 ▷ Generate an admissible 5B symbol
            $retval\_DSs[i] \leftarrow$ GET_RANDOM_ADMISSIBLE_5B_SYM
        end if
    end for

    $rb =$ GET_RANDOM_BIT

    return PLACE_DSS($retval\_DSs$, $rb$)
end function

procedure TEST_MAIN
    ▷ Algorithm exhaustively testing the {19,19} coding scheme
    ▷ Note: to get reproducible results, a constant seed for the TRNG is preferred
    for $s \leftarrow 0$ to $5^{15}$ do
        ▷ $\_DSs$: a (19,17) RSC codeword having only
          the 76 bits (MSs) coming from the DSs
        $\_DSs \leftarrow$ GENERATE_76_DS_BITS($s$)
        ▷ $\_DSs\_and\_SBs$: RSC codeword DSs and SBs
        $\_DSs\_and\_SBs \leftarrow$ ENCODE_SCHEME($\_DSs$)
        ▷ $s1$: channel input, as per Figure 3.2
        $s1 \leftarrow$ ENCODE_RSC($\_DSs\_and\_SBs$)
    end for

    if COUNT_FSS($s1$) $\neq 0$ then
        EXIT(False)
    end if

    for $e \leftarrow 0$ to 20 do
        $s6 \leftarrow s1$
        ▷ When $e = 0$, do not induce an error, this way testing the error-free
          transmission as well
        if $e \neq 0$ then
            $s6[e-1] \leftarrow s6[e-1] \oplus$ GET_RANDOM_ERROR
        end if
        if $s1 \neq$ DECODE_SCHEME($s6$) then
            EXIT(False)
        end if
    end for

    EXIT(True)
end procedure
```