Hamming Codification for Safety Critical Communications

Faculty of the Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona (ETSETB), Universitat Politècnica de Catalunya
Section of Machine Interlocks (TE-MPE-MI) , European Organization for Nuclear Research
by

Olivia Jullian Parra

In partial fulfillment
of the requirements for the Master in
ADVANCED TELECOMMUNICATIONS TECHNOLOGIES

Advisors:
Dr. Beatriz OTERO, UPC, Spain
Msc. Iván Romera, CERN, Switzerland
Barcelona, June 2022

Contents

List of Figures

Listings

List of Tables

Abbreviations

ALICE A Large Ion Collider Experiment

ARQ Automatic Repeat Request

BDS Beam Dumping System

BIS Beam Interlock System

CERN European Organisation for Nuclear Research

CIBM Beam Interlock Controller Manager

CMS Compact Muon Solenoid

FECC Forward Error Control Coding

GUI Graphical User Interface

HDL Hardware Description Language

IO Input/Output

LHC Large Hadron Collider

LHCb Large Hadron Collider beauty

LUT Look Up Table

MPS Machine Protection System

RAM Random Access Memory

SFP Small Form-factor Pluggable transceivers

UTC Coordinated Universal Time

VHDL Very High Speed Integrated Circuit Hardware Description Language

VME Versa Module Europa bus

Abstract

The Large Hadron Collider (LHC) at the European Organisation for Nuclear Research (CERN) is one of the largest particles accelerators in the world. Due to the complexity when colliding these particles at high energy and the high cost of failure (in both financially and efficiency aspects), a Machine Protection System (MPS) is requiered, monitoring (CERN) high energy accelerators and protecting all parts of the accelerator when there is beam presence.

This backbone of the MPS relies on a Beam Interlock System (BIS). The BIS transmits the request from any equipment system in the MPS to either a Beam Dumping System (BDS) (to eject safely the particles from the accelerator) or inhibits the injection of the beam to the accelerator. Thus, the communications between BIS elements are declared as safety critical communications.

The purpose of this thesis is to explore different communication protocols that could be used to send and receive data between the systems that compose the BIS. The current method used is Manchester modulation, which encodes data achieving a zero overall DC bias. Besides its simplicity, as this method incorporates clock-matching between the transmitter and receiver devices within the data stream itself, the bit rate is essentially halved, limiting the protocol itself.

This thesis compares and contrasts the current approach with Hamming codification. Results show that the data can be encoded with a similar resources efficiency without the necessity of clock matching at the receiver, as well as a zero overall DC bias. The conclusion guides then to an improvement of transmission length and reliability confidence between these elements.

Revision history and approval record

Revision	Date	Purpose
0	19/02/2022	Document creation
1	10/06/2022	Document implementation
2	17/06/2022	Document revision

DOCUMENT DISTRIBUTION LIST

Name	e-mail
Olivia Jullian Parra	olivia.maria.jullian@estudiantat.upc.edu
Beatriz Otero Calviño	botero@ac.upc.edu
Iván Romera	ivan.romera.ramirez@cern.ch

Written by: Olivia Jullian Parra		Reviewed and approved by: Beatriz Otero Calviño	
Date	10/06/2022	Date	17/06/2022
Name	Olivia Jullian Parra	Name	Beatriz Otero Calviño
Position	Project Author	Position	Project Supervisor

1 Introduction

Nowadays, the LHC at CERN is the largest and most powerful particle accelerator in the world. It consists of a 27 kilometre ring of superconducting magnets that boost the energy of the particles along the way.

The accelerator is composed of two high-energy particle beams (a series of bunches of particles). Travelling at almost the speed of light and in opposite directions, the collision of these two beams are used for four different particle detectors: ATLAS [1] (to search for the Higgs Boson [2] and particles that could potentially make up dark matter), the Compact Muon Solenoid (CMS) [3] (to also search for dark matter particles), for A Large Ion Collider Experiment (ALICE) [4] (designed to study the physics of strongly interacting matter at extreme energy densities) and the Large Hadron Collider beauty (LHCb) [5] (to study the differences between the matter and the antimatter).

Colliding particles together with very high energy (6,8TeV per beam) needs a specific MPS to protect all parts of the accelerator complex that might handle beam above damage thresholds. This MPS relies mainly on the BIS, which has the power to remove the beam from the LHC when the machine encounters a problem that could pose a risk to the accelerator and its attached systems. Since this system is at the heart of the MPS, this project is based on the requirements of the BIS elements. Figure 1 shows an overview of the BIS.

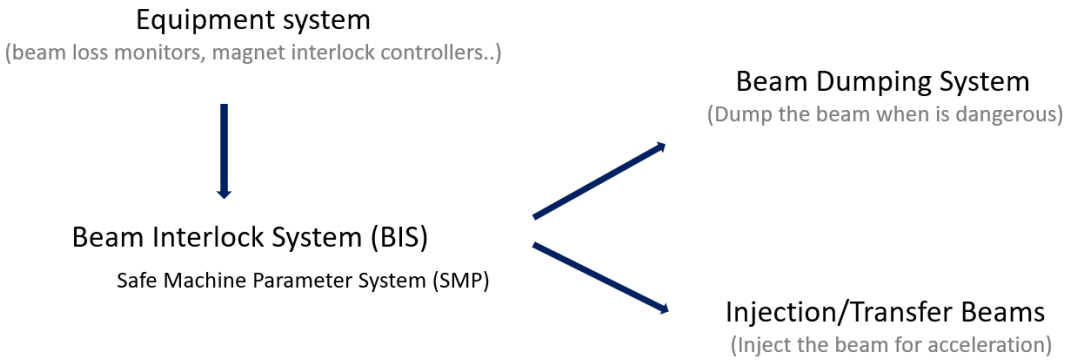

Figure 1: BIS overview

Diving into the BIS, this system takes signals (called User Permits) from various user systems (located in the accelerator and monitoring the beam characteristics) and converts them into permits (called Beam Permits) used by the injection, extraction and BDS to allow or not beam in the accelerator. B. Todd [6] presents the different elements or systems that consolidate the BIS and details the main BIS purpose. In [6], we can see that there is a reaction time of 100us for communication between the BIS and the BDS. This represents the overall time since the user sends its request until the BDS is ordered to either extract or not the beam from the accelerator (representing the dump of the beam).

Nowadays, the interaction of the elements of the BIS is done for the test and monitoring of the user systems connected to the BIS itself. Thus, the communication between the

elements in the BIS must be not only reliable (due to the goal of a machine protection system) but as well fast (to detect on time possible user problems). It is for that reason that a simple codification (Manchester modulation with a single parity check [6]) is the current method used.

Manchester modulations are self-clocking. Thus, they incorporate the clock into the actual data signal itself, meaning that each bit transmitted will contain one rising edge or one falling edge (if there is no edge in the bit, the state is illegal and will be ignored). This is a very useful trait in ascertaining useful bits, though of course at the expense of speed.

Therefore, an appropriate clock in the receiver part provides a simple and fast decryption of the data. However, a slight drift on the frequency of the clock can lead to a large number of errors, rendering the signal useless, as the clock is not re-matched to the signal at any point during transmission. Thus, the efficiency of this codification is dependent on the sampling of the data and on the drifting of the clock. This makes Manchester modulation inoperable at high data rates, and prone to error for big data transmissions.

As an example, liked to the BIS, the data rate used is 62,5kHz. Considering an ideal scenario, where the clock does not drift at all, Manchester can be considered suitable for transmission, as the transmissions are not considered high speed communications. But, as far as there are errors in the frames, using a single parity check and a code violation [6] (error detected from a bad demodulation of the frame) results inefficient. The reason why this is inefficient is that the entire data stream must be transmitted again, as the decoder has no way of knowing where the error occurred. On the other hand, due to the limitation of big data transmissions, the current approach only allows the codification of one byte of data, as in, one parity bit each 8 bits. However, [6] shows how the most common transmission is 32 bytes.

In this work, we address those limitations in two parts. The first part of the project ensures the correction in real time of single bit errors during transmissions, ensuring always a valid rate of 62,5kHz (only achieved in the current system considering an ideal scenario). The second part of the project offers the possibility of an efficient transmission of at least, 5 bytes of data.

With the division of the project in two parts, the following contributions are provided:

- A comparison of different safety critical communications protocols.
- The simulation of the best alternative protocol found for safety critical communications.
- A practical comparison between Manchester modulation and Hamming codification for a specific use of case: the BIS.
- The overcome of the current protocol limitations presented above.

2 Methodology

After introducing the main goal of this project (find alternatives for safety critical communication's protocols), different procedures and analyses are performed to achieve the desired goal.

Due to the complexity of the systems involved in the project, targeting the problem and presenting the use case is essential to understand the requirements of the solution proposed.

Then, once the current protocol and its limitations are exposed, the objectives of this project are presented. A survey on the different codifications is done to find the best alternative. The research of related works are also done to understand the limitations of the project and their alternatives. The performances of the current BIS communications are used to interpret the results of this project.

As explained in the previous section, the deployment of a communications protocol for safety critical communications is not trivial. This project considers this complexity and divides the project in two parts: the design and simulation of a Hamming codification for the BIS communications, and its deployment in a use case scenario.

Regarding the implementation and simulation, different experiments are done: first, only the communication's protocol is simulated and then, it is used in a real scenario, implementing a single self loop communication in one of the systems.

After the realization of the simulations, a comparison between Manchester modulation and Hamming codification is done to present the main differences and possible limitations of both protocols.

The results of the simulations are used to ensure that the limitations encountered above are overcome sufficiently. Once the new protocol has been simulated and compared with the current one, the new protocol will be deployed and tested in working hardware. Here, the limitations of the project are detected and propositions of solutions are presented.

Finally, the results of the execution are interpreted, demonstrating an effective solution for safety critical communications. In order to delve deeper into the different procedures of the project, the work plan is explained in Appendix A.

3 Safety Critical Communications: Current Codification

The current communication among the elements of the BIS is done using Manchester codification with a single parity bit check. Manchester encoded frames use a two bit edge based encoding, with a rising or falling edge representing a transmitted logic '0' or '1'([6]), as figure 2 shows.

Using the rising or falling edge for a bit representation reduces the link bandwidth to half of the encoding frequency. On the other hand, since the protocol is based on clock

Figure 2: Manchester modulation for logic '0' '1'

drift (as presented in the introduction), it used for clock recovery and robust decoding (allowing the detection of a single bit error during transmission).

The frame to transmit is composed by a Code Violation (effectively represented by 3 bits; "000", that have not been encoded in Manchester) that allows framing synchronisation. This tries to admit certain tolerance to possible drifts in the clock frequency of the data transmission.

Once the code violation, "000", has been transmitted, a preamble is sent along with a start bit. The Preamble comprises of three logic 1s, encoded in a Manchester format, so "010101", and the start bit, a single logic 0, so encoded as "10". These four bits , eight bits once encoded, represent the start of a new frame.

Following this, the proper codification of the payload starts. The sender is allowed to send between 1 and 4 bytes of data without repacking the Code Violation, Preamble and start bit, but the Manchester modulation with single parity check (and thus the protocol to be analysed) is done for each byte.

The main idea is to calculate an even parity for each byte and modulate the result using Manchester modulation. Thus, between bytes we will have an additional even parity check.

3.1 Codification

The encoder is based on a shift register, introduced by the user. As mentioned before, the user is able to transmit between one and four bytes of data. Finally, the used frequency is 62,5kHz.

Finally, for each byte, the protocol calculates the even parity (XOR of all the elements in the byte) and modulates the resulted nine bits. Then the information is loaded in the shift register (and shifted out one at a time when it is transmitted) when a flag goes high (ready to transmit) and a second flag is set when the shift register is empty, allowing the introduction of new information. An example can be found in figure 3.

3.2 Decoding process

The decoder aligns its clock with the transmitted data, samples the payload, and validates the correct reception of the frame received. This is done using a state machine. The number of expected bytes to be received is set by the user. This parameter, along with the encoding

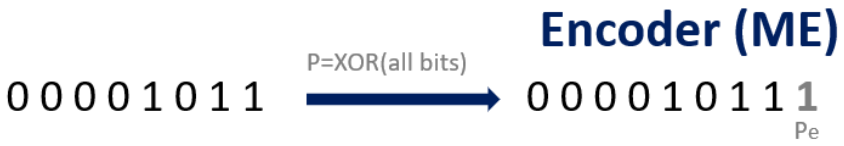

Figure 3: Manchester encoder

frequency is known and declared when designing the systems of the BIS and therefore can be declared beforehand.

The start of the decoding process is a simple check of the Code Violation, Preamble and start bits. During the check, the clock synchronises to sample in the center points of the incoming data (in order to decode efficiently).

When the sampler has been correctly synchronised, it stores the incoming bits into a shift (bit by bit) register. When the register is full (meaning we have received all the payload data), the demodulation for each nine bits is done to detect code violations (in this case a single bit error detection).

Finally, after the demodulation, the recalculation and comparison of the single parity check every eight bits is done to detect payload errors (single bit error). If the validation is successful, a flag is set to indicate the data is valid. Otherwise, an error flag is set and the receiver will wait for a re-transmission of the entire payload.

Please note that, for reliability purposes, in the BIS elements, even if the rest of the bytes are correct, detecting an error will discard the entire payload.

Figure 4 shows an overview of the current decoder:

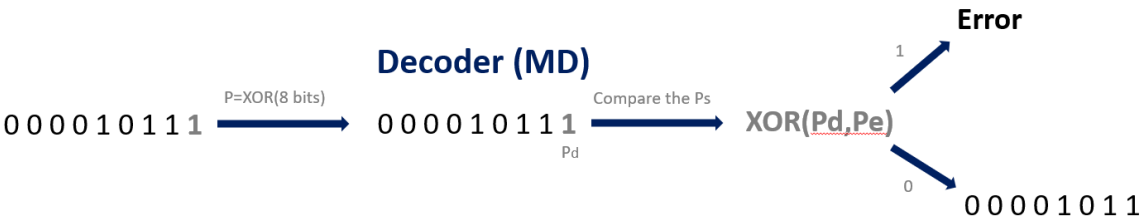

Figure 4: Manchester decoder

3.3 Limitations

Once the current system is presented, the following limitations have been detected:

- When a code violation is found in the decoder (single bit error detection), the received frame is discarded, leading to the loss of the entire payload.
- Due to the lack of single bit error correction, the re-transmission of the payload is expected when a code violation is detected. This leads to consider the rate transmis-

sion of 62,5kHz as an ideal scenario and thus, not representing entirely the reality for the communications among the BIS elements.

- A segment of the payload is required to codify the transmitted frame correctly, having an inefficient protocol for big data transmissions.

In the following sections, these limitations are considered to first: search for new alternatives to Manchester modulation and single even parity check, and second: deploy and compare in a use case scenario the best alternative against the current protocol.

4 Related Work

This section describes the different techniques applied for safety critical communications. Literature is used to support the best approach for the design of the communications protocol developed in this project.

4.1 Techniques for Safety Critical Communications

Safety critical systems are complex systems often combining physical and mechanical components, networks, hardware and software. Those systems rely on safety critical communications, where all the information transmitted is seen as potential source of failures (as it is considered on the BIS elements).

To make the information transmissions more efficient and reliable, many codifications techniques have been proposed. Wang et al. [7], present a review of encoding-decoding techniques and their limitations for different types of networked systems.

Considering the requirements of the BIS elements, we will focus on channel coding techniques, also known as Forward Error Control Coding (FECC). FECC focuses on detecting and correcting bit errors in digital communications systems. Faruque et al. [8], present the principles and realization of those coding techniques. More detailed information is explored in [9].

The current technique used for the BIS elements is based on the Automatic Repeat Request (ARQ) technique. This technique adds parity bits used by the decoder to detect an error in the received data. After an error is detected, the decoder waits for the re-transmission of the entire data again. Although the ARQ technique bases its re-transmission in acknowledgements, the BIS elements use error flags for the same purpose.

On the other hand, in a FECC coding, data is encoded with redundant bits to allow the receiver to not only detect errors, but to correct them as well. The main goal of all FECC techniques is to detect and correct as many errors as possible without increasing the data rate or the bandwidth.

As we can see, FECC techniques will be the desired ones for this project since its objective is to overcome the limitations of the ARQ and thus, the ones presented in the current system.

Figure 5 shows a scheme of the different techniques explored in this section for the deployment of a new safety critical communications protocol.

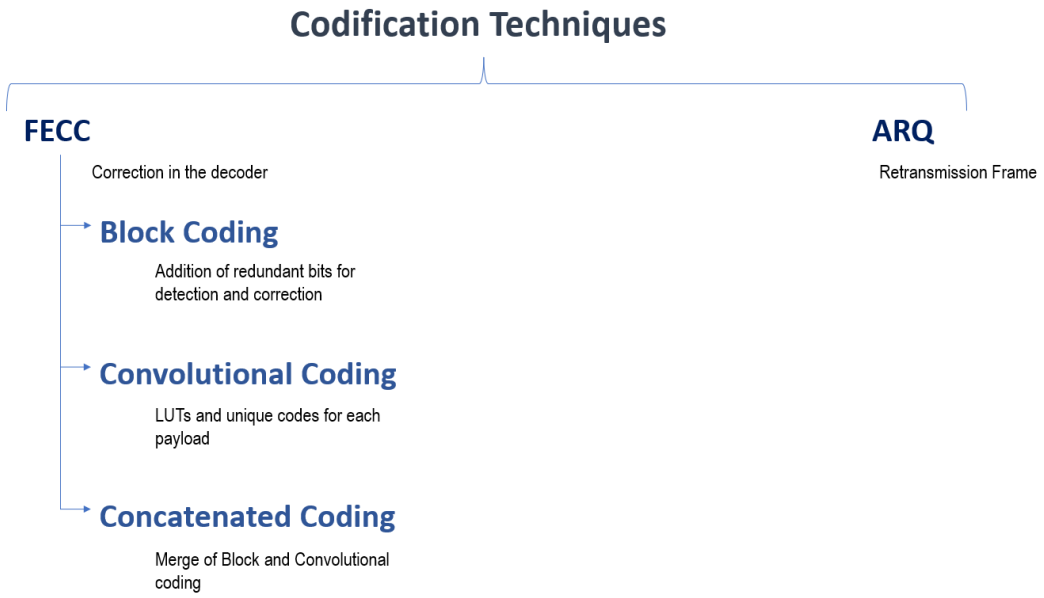

Figure 5: Codification techniques for safety critical communications

4.2 FECC techniques for Safety Critical Communications

Many different FECC techniques can be suitable for our project. Here, an overview of the principal ones is done and the selection of the most suitable is deployed and compared in the next sections.

4.2.1 Block coding

In block coding, bits are segmented into blocks of m-data bits. The encoder transforms each m-bit data block into a larger block of n-bits, called code-word, where $n > m$. The difference $(n-k)$ bits are the redundant bits r, also known as parity bits. These redundant bits do not carry information, but enable the detection and correction of errors. [10] details this codification. Hamming codification is an example of block coding.

4.2.2 Convolutional coding

The main idea of convolutional coding is that m information bits enter in the convolutional encoder sequentially, where the r redundant bits $(r > m)$ are generated through shifts of the sequential input bits. After calculating the parity bits, a Look Up Table (LUT) is produced to store all the unique codes for the m payload bits (2^mcombinations). Then, these encoded bits are modulated and transmitted through a channel. At the receiver side, the receiver decodes by means of code correlation and regenerates the information bits. More information can be found in [8] and [10].

4.2.3 Concatenated coding

We can classify concatenated coding in series concatenated coding or parallel concatenated coding. For simplicity, only the simple coding is considered in this comparison, but more information can be observed in [10].

Series concatenated coding is excellent in error control. Here, the data is first encoded by usually a block code, forming the outer code. Next, the data and parity bits resulting from the outer code are interleaved and encoded by a convolutional coding, called the inner code. The final code-word is then modulated and transmitted.

On the receive side, we first decode the inner coder and the outer. The main advantage of the concatenated coding is that any errors which do not get detected by the inner code are corrected by the outer code.Thus, reliability, in this coding technique, is paramount.

4.3 Final comparison

Table 1 shows a comparison between the FECC techniques considering reliability and efficiency aspects. Information has been extracted from [11],[12] and [13].

Table 1: Techniques for safety critical communications

Reference	Technique used	Complexity	Efficiency	Time
[11]	ARQ	simplest	least efficient	time consuming
[11], [13]	Block coding (Hamming)	simple	efficient for single and double bit flips	good for low latency
[12],[13]	Convolutional coding	complex	efficient	intermediate
[12]	Concatenated coding	most complex	most efficient	time consuming

As we can see, Block coding techniques are the simplest, most reliable and most efficient techniques considering BIS purposes. Simplest because it requires little modifications from the current system deployed and most reliable due to the purpose of single bit error correction, adding the possibility of double error detection. Finally, even though there are techniques more efficient for error detection, considering the purposes of the current system, the main idea is to consider the simplest codification that allows real-time error correction of single bit flips.

Thus, a Block coding will be deployed and compared to the current ARQ (Manchester modulation with single even parity bit) technique. Therefore, due to its simplicity, Hamming codes will be used for this project.

5 Approach Proposed: Hamming Codification

The final approach is proposed as a result of a comparison between the different codifications proposed in section 4. The principal benefit of Hamming codes is that it is able to correct single errors; this represents a great improvement in the system as a whole overall, as a major limitation of the current deployed solution (Manchester codes with

a single parity check) is that it is unable to correct single errors. In addition, Hamming offers double error detection, increasing the robustness of the communication.

Hillier et al. [14] gives an overview of a Hamming codification, applying its principles to on board nanosatellites due to the advantages in terms of memory, resources and computational cost that a Hamming solution offers for critical communications. Figure 6 shows the Hamming codification overview that will be used in this project:

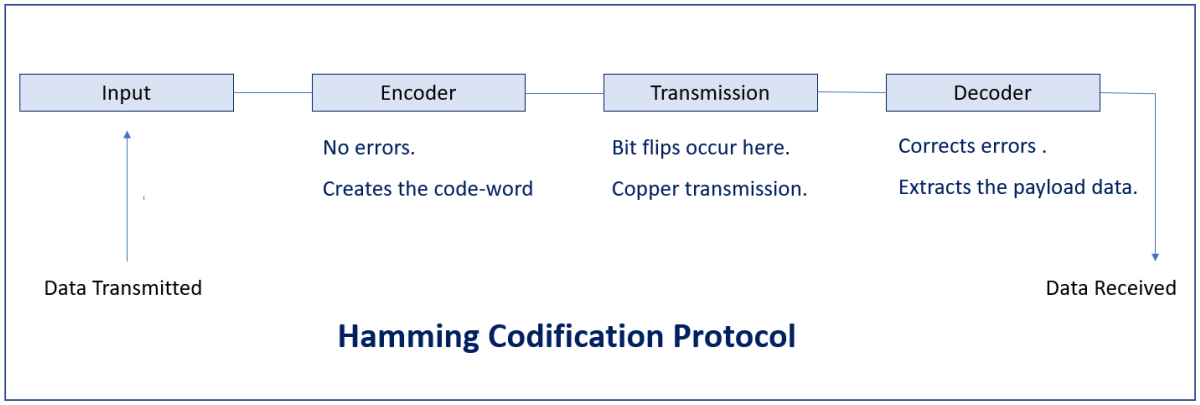

Figure 6: Hamming codification protocol overview

From Hiller et al. perspective [14], Hamming codification is based on k clean input bits (without errors). This payload is sent to the encoder who will add r redundant even parity bits (also called syndromes) allocated in positions power of 2 (for instance 0, 1, 2, 4, etc.) and calculated using Hamming theorems [15]. The attachment of these parity bits to the payload and the addition of a final single even parity bit check (XOR of all r + m bits of the encoder) forms a code-word of n-bits that will be sent to a decoder in the receiver part.

Errors are prone to happen during the transmission of the data (after its codification) or when storing it (before decodification). These errors are defined as bit flips in the memory or storage. Following the Hamming codification, a single bit flip is corrected and, by adding the extra single even parity check, a double bit flip is detected.

The decoder will be responsible for detection and correction. To achieve that, the decoder will apply again Hamming theorems [15] to calculate the redundant even parity checks and calculate the syndrome (also explained in the subsection below). Finally, the decoder locates and corrects (if single bit flip) the error contained in the code-word before extracting the payload (m bits).

Overall, the implementation of this protocol may require more memory for small payloads and thus, more clock cycles to be implemented. In the following sections though, it is theoretically proven that, with the increase of data transmitted, there is also an increment in the efficiency of the code. On the other hand, due to non-modulation of the logical '0's and '1's, the transmission time is decreased to almost half than the time used with Manchester modulation. As the efficiency increases with the transmission of data, we

can see from the codification subsection that, for more than 7 bytes of transmission, the decrease in time represents more than 50%.

As a result, Hamming codification is faster, more robust and more efficient than Manchester modulation with a single even parity check bit for copper based transmissions (which is the use case of the BIS).

The following subsections will provide a deeper explanation on the Hamming Algorithm for codification.

5.1 Codification

The Hamming encoder is responsible for, given a payload information of m bits, generating the code-word of n-bits explained above. Many different steps have to be followed to get the final code-word: first, the number of redundant even parity check bits r has to be calculated with the following formula:

$$2^r \geq m + r + 1$$

where m is the number of payload bits and r the number of redundant even parity check bits.

After knowing r and following the Hamming theorems [15], the even parity check bits are allocated in their correspondent positions (shown in algorithm 1) and, after allocating in between the m payload bits, calculated based on the following statements:

- Each of the redundant even parity check bits represents a position Pi: 2 to the power of parity bit (P1 for bit 0, P2 for bit 1, etc).
- The position of each redundant even parity check bit in the n-bits code-word designates the payload bits considered for its calculation: for instance, P1 considers for its XOR logic door all the sets of 1; thus, all the odd numbers are considered for its calculation. Following the same statement, P2 will consider all sets of 2 (3, 6, 7, etc.), P4 sets of 4 (5, 6, 7, etc.), etc.

Finally, the last parity even check is calculated with a XOR logic door of all the elements of the code-word (the parity bits and the payload). An overview of the encoding algorithm can be found in Algorithm 1:

Algorithm 1 Hamming Encoder Algorithm

1: Calculus of r (number of redundant even parity check bits): P1,P2,...Pr.
2: Allocation of the redundant even parity check bits in position: 2 to the power of parity bit (1, 2, 4, 8, etc).
3: Allocation of the payload (m bits) in the positions between the redundant even parity check bits.
4: Calculation of each redundant even parity check Pi.
5: Calculation of the single even parity check additional (XOR(m+r bits)).

5.2 Decodification

Once the bits are encoded, the transmitter will send the frame to the receiver. It is while sending and storing the data that the communication is prone to errors. Therefore, the Hamming decoder is responsible for again generating the syndrome (the r redundant even parity check bits) from the n-bit code-word received. To achieve that, the decoder will first detect the m-bits of payload and then recalculate the syndromes (r-bits plus the last parity check for double error detection).

Once the syndromes are recalculated, the XOR of the new syndromes and the received ones will lead to a vector c'. The natural number represented by this vector, leads to a position c. This position is calculated as it follows:

$$c = \sum_{i=0}^{r} 2^i * c'_i$$

Combined with the final even parity check p, we can detect and correct errors following Table 2:

Table 2: Detection and Correction of Errors with Hamming Codification

c Value	p Value	Descrition
0	0	No errors
0	1	Error in p
!=0	1	Single Error in position c
!=0	0	Double Error detected

Once the errors are detected, if only a single error is detected, the decoder will correct it by using a XOR logic door between a logic 1 and the value of the code-word bit at position c.

5.3 Limitations overcome

The limitations of the current protocol explained in section 3 are crucial for safety critical communications. After the proposal of this new method, we conclude that Hamming codification will allows us to overcome all those limitations adding in addition, a new level of robustness.

The reason resides in the possibility of single bit error correction, avoiding the re-transmission of the payload after an error is detected and thus, making real the transmission rate of 62,5kHz. As an addition, a double error detection is possible, adding redundant security on the transmission.

Finally, looking at the redundant parity checks calculation algorithm, Hamming codification allows the same robustness for different data payloads. This overcomes the inefficiency of Manchester modulation with single even parity check for big data transmissions.

In the next sections, we will describe the design of the protocol, as well as its simulations and deployment for the BIS use case scenario and a final comparison with the current Manchester modulation, n order to ensure that the limitations, presented in section 3, were overcome .

6 Hamming Codification Design

Since the communications presented in [6] are deployed in hardware elements (BIS cards), the posterior implementation of the design presented below is done in Very High Speed Integrated Circuit Hardware Description Language (VHDL).

The following subsections will describe the design used for both encoder and decoder Hamming communication.

6.1 Encoder design

Due to the characteristics of the BIS elements [16], a state machine is designed for both Encoder and Decoder. Figure 7 shows the block diagram the Hamming Encoder design.

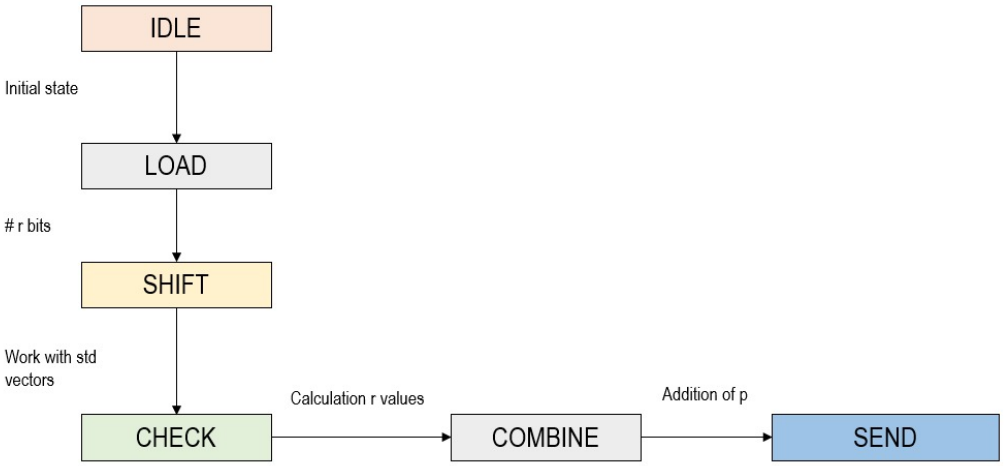

Figure 7: Sate machine for Hamming encoding

As we can see, the encoder is composed by the following states:

- IDLE: Initial state. It indicates the end and start point of the Encoder.
- LOAD: Calculation of the r number of parity bits (called also syndromes). Declaration of a standard logic vector as the final code-word of n-bits. Loading of the data bits into a single standard logic vector. Insertion of the r syndromes in the code-word. The bits are set at position 2 to the power of parity bit (1, 2, 4, 8, etc) and with value 0.
- SHIFT: Shift to the right the code-word standard logic vector in order to adjust the standard logic vectors (instead of declaring a vector from 1 to n standard logic vectors are declared from 0 to n-1).

- CHECK: Calculation of the syndromes as explained in section 5.
- COMBINE: Calculation and addition of the single even parity check p.
- SEND: Transmission of the data bit by bit and at 65.2kHz as required in the BIS.

Using a state machine allows us to implement the Hamming codification following very closely the Algorithm 1. This also allows the future user of the codification to have a better understanding of the process without losing efficiency.

6.2 Decoder design

The decoder is designed using the same method as the encoder with the following states:"

- IDLE: Initial state. It indicated the end and start point of the Decoder.
- LOAD: Copy of the received standard logic vector. Set to 0 of the syndromes and final even parity check is done to recalculate them on the next state and compared them afterwards.
- CHECK: Calculation of the syndromes as explained in section 5.
- DETECTION: Comparison of c and p. Detection of errors following Table 2.
- CORRECTION: If there is a detection of a bit flip, do XOR of the payload bit in position c and a logic 1.
- PACK: Extract the data from the code-word vector and set a flag if a single or double error is detected.
- BYTE ARRAY: Repack the payload in bytes.
- SEND: Transmission of the data to memory storage.

Figure 8 shows the overview of this final state machine.

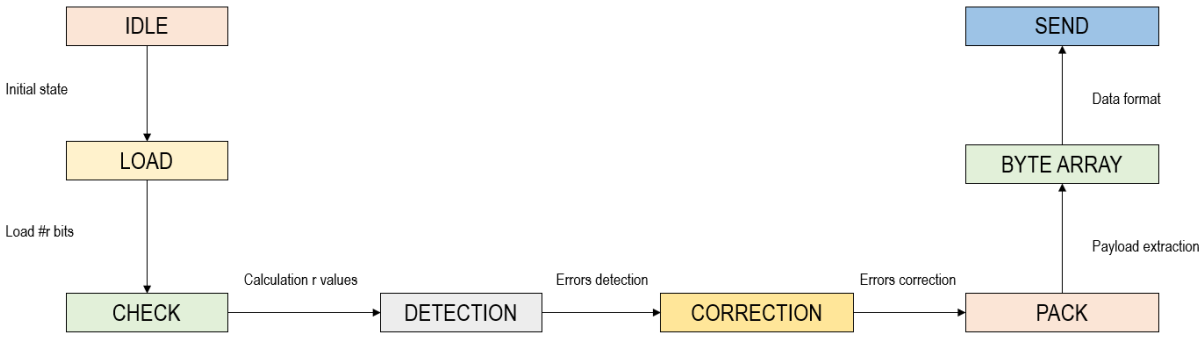

Figure 8: Sate machine for Hamming decoding

In the next section, the deployment of the design through first, simulations and then, a practical use case in one of the elements of the BIS is explained. The setup of the experiments, especially the practical use case, is also presented in section 7.

7 Experiments

Once the design of the new protocol is defined, this section uses one of the elements present in the BIS elements to test the new codification in a practical use case scenario. The simulations of the designed algorithm, will guide us to a first understanding of what the posterior experiment shows.

7.1 Setup of a self-loop communication

As explained in section 6, Hamming codification is designed following the same requirements used in [6]. Therefore, in this section, VHDL will be used in Vivado for both simulating the solution, and providing the practical use-case that will be deployed in the final system. Vivado is a software suite produced by Xilinx for synthesising Hardware Description Language (HDL) designs, and also has the ability to simulate designs. A general protocol module is used for the implementation of both codifications. The module is composed of the following parts:

- TOP: A module used to deploy a self-loop communication. In there, the module TRANSMITTER waits for data to transmit and sends it, bit by bit, after encoding all the payload using the corresponding TRANSMITTER submodule. Then as a self-loop behaviour, the data is decoded using the RECEIVER module (different as well as the TRANSMITTER submodule for each of the codifications).
- TRANSMITTER: A module used to encode and transmit data. This module codifies the data to transmit and sends it, bit by bit at a specified frequency. Composed by the ENCODER module (different for both codifications).
- RECEIVER: A module used to decode and store data. This module synchronizes the clock with the TRANSMITTER module. For that, it checks the initial preamble bits to ensure a good synchronization. Then a DECODER module (also different for both codifications) is used for the decodification of the received data.

After understanding the structure of the self-loop communication, we can notice that, for each codification, the module content is different but the design structure remains the same (both have TOP, TRANSMITTER and ENCODER, and and DECODER). Since both modules follow the same structure, only by changing the encoder and decoder submodules and their corresponding port maps in the transmitter and RECEIVER submodules, either protocol can be used, not only for simulations, but also for real case scenarios

7.2 Simulations

In order to understand and ensure a good communications protocol in a practical use case, before experimenting the self-loop communication with the hardware used in the LHC, a simulation is done using the Vivado simulator [17].

The simulation is compiled for a Field Programmable Gate Array FPGA from Xilinx, a Spartan 7. The reason resides on the posterior experiment. There, a Spartan 7 acts as both transmitter and receiver of BIS data transmission.

Different signals are considered in the simulations. The most relevant are: the clock (working at 62.5kHz) in channel 0, the clock cycle of 500us in channel 1, the number of bytes transmitted in channel 3, the desired transmission data in channel 4 , the decoded data in channel 6, the flag for single bit error correction in channel 7 and finally, the flag for double bit error detection in channel 8.

Considering both simulations, a comparison between the simulation for both Manchester modulation with a single even parity check and Hamming codification can be done. As expected, using Hamming codification, results in a decrease of almost 50% of the transmission time after the first byte of transmission. Section 8 discusses in detail the results extracted by these simulations.

In addition, Figure 9 shows the possibility of double error detection when using Hamming codification highlighted in red).

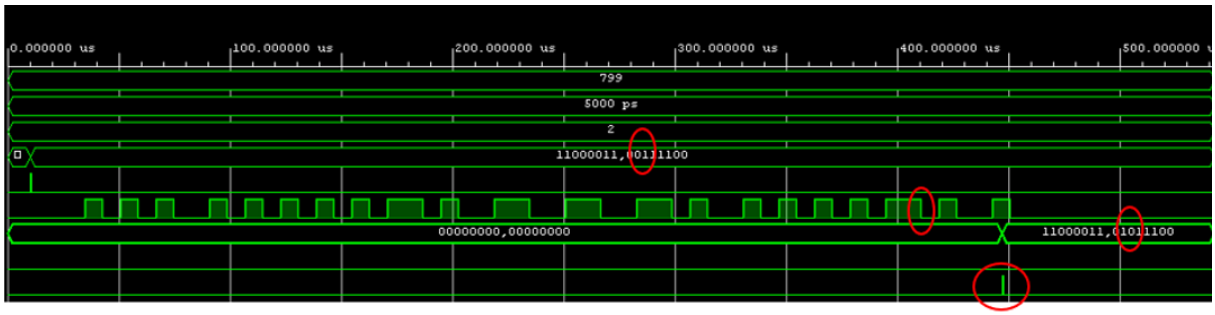

Figure 9: Double error detection with Hamming codification

Finally, as explained before, the structure of the general module for both communication protocols reminds the same. This sets the design of a Hamming codification maintainable and easy to implement in real BIS elements. As a result, in the next subsection, an example of one of the use cases of the BIS is tested with the new protocol deployed.

7.3 Practical use case: BIS communications

Once the simulation of the new codification is tested, the setup of the practical experiment is presented in this section. Since the goal of this project is based on the communications among the BIS elements, we must deploy the self-loop communication in one of the cards that composes the system: the Beam Interlock Controller Manager (CIBM).

The CIBM [16] is considered to be the heart of the BIS system. The reason resides in the goal of the card itself: the CIBM is the card that, given a set of user permits, transforms them into beam permits. This means the CIBM controls and monitors the requirements of the system to either dump the beam, or allow the beam to remain in the machine

Since all the critical communications will be managed by this card, the experiment is deployed there, externalizing the self-loop communication through a copper bridge between one input and one output of the CIBM itself.

The implementation of a self-loop communication is done for convenience and practicality: since the simulations are based on self-loop communications, adapting it to a real case scenario is easy and fast (as explained above). To compare both protocols, deploying a self-loop communication has no impact nor difference compared to a deployment based on a communication between two different elements, since the setup is externalized from the CIBM through a backplane, and reintroduced sing a pair of copper cables, as the connection is differential. The externalization and use of a copper bridge, allows us to represent transmissions between different cards while only using one card.

Therefore, with the deployment of a self-loop communication in the CIBM, the practicality and use of Hamming codification is demonstrated and can be considered for future updates of the BIS system. In order to test the codification in a real environment, the set up and results are shown in the next subsections. Results and comparison with the current system of telecommunications are detailed explained in section 8.

7.3.1 Set up

The CIBM is responsible for the analysis and control of the user requirements (inputs received from elements in the accelerator) and the decisions of the future of the beam (outputs given to the BDS or injectors). Having Inputs and outputs allows us to set up a self-loop communication using only one card, only adapting the simulations modules.

In order to recreate a communication between two elements of the BIS, only one input and an output of the CIBM is used. For reliability purposes (robustness in the communications), the inputs and outputs of critical signals are differential. This means that, for each bit transmitted, two (one with the same value and another one with the opposite) are sent to the exterior of the card through its back-plane. With the same purposes, each input is also differential.

For this experiment, the Spartan 7 FPGA of the CIBM is programmed to send to one of the outputs 5 bytes of data. The sent data is transmitted with a copper cable to a differential input. Then the communication is monitored through high level scripts (python Graphical User Interface GUI-based application) and a logic analyzer.

Figure 10 shows the setup of the experiment.Details of the experiment and the results acquisition are explained in the next section.

7.3.2 Experiment

Once the copper cable bridge has been connected between the input and output of the CIBM, the implementation of the experiment in a real environment can start.

The experiment consists on the transmission to a defined address (1 byte of data) of 4 bytes of payload data. The payload decided (5 bytes of data), as well as the address set is done based on the actual system implementation ([6],[16]). Both address and data are set by the user through a python-based script that has access to the Versa Module Europa (VME) of the CIBM. Once the data is set by the user (through a GUI), it is constantly sent to the output and received from the input of the CIBM.

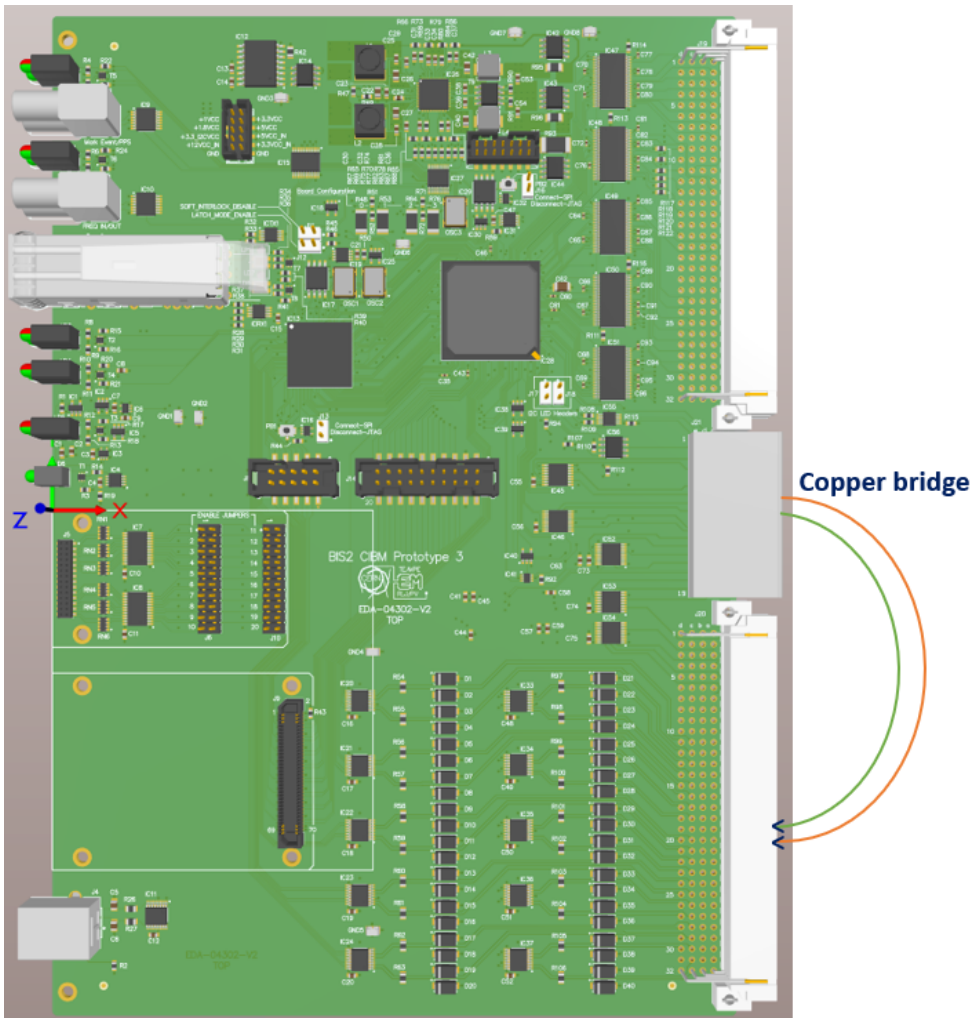

Figure 10: Set up of the self loop communication

The modules used by the simulations (both TRANSMITTER and RECEIVER) are used to encode, transmit, receive and decode the data set by the user. The FPGA contained in the CIBM (a SPARTAN 7) is responsible for the data transmission, reception and analysis. Thus, when the receiver triggers and error (setting a flag), the FPGA stores the event on a history buffer.

The history buffer records certain pre-set information that is defined at synthesis; when a certain event occurs, the details of that event are stored in a large Random Access Memory (RAM) block that records what triggered, at what time, any ancillary information concerning the trigger, and any additional information that might be useful to the user to find out what happened and why. Once an event is stored, it remains until it is overwritten. The history buffer used in this experiment is 2048 rows long, so all 2048 rows must be written to before past rows are overwritten with new records.

There can be different types of events registered in the history buffers of the CIBM: communication errors, data updates, timing errors, etc.

To monitor communication errors stored in the history buffers every time new data is sent, we first need to know which is the last recorded history buffer entry (stored in the FPGA register HB_DIAGNOSTIC_LR).

This register will tell us the offset (in decimal) needed to access the stored event in the history buffers. The history buffer record is stored in the next available record space. If all records have been written to, the record will overwrite the next row, as the address map increments by one each time a new record is written.

Once we have the entry for the desired event, the following 4 registers will show information about the stored action. Each of the registers is 32 bit long and contains the following information:

- Timestamp in Coordinated Universal Time UTC seconds.
- Timestamp in UTC microseconds.
- Type of event: number of records left (position in the history buffer registers) + ID of the record. This register will tell us which type of error/event has been recorded.
- Details about the error/event.

In this experiment, as there is a new protocol that is able to detect double errors, a modification of the current CIBM history buffer is done. The new history buffer includes then a flag for double error detection. Table 3 shows the different communications errors stored in the history buffer of this experiment and their codes.

Table 3: Communication errors stored on the History Buffer

Type of error	Code
Single Error Correction	0101
Time out	0110
Double Error Detection	0111

In this experiment, hard coded errors are introduced to ensure their detection and correction. Their introduction is done by adding a new state in the state machine of the encoder module. Figures 11, 12 and 13 show the detection and correction of a single bit error using Hamming codification.

In Figure 11 we can see the analysis of the communication using a logic analyser. Different tests points are introduced in the signals of the CIBM FPGA and analysed by the logic analyser. Among them, channel 0 represents the data sent by the transmitter. This channel shows that the errors are effectively hard coded, since channel 1 is the data received and both channels show an error in the first payload bit. The error can be found if we compared to the data sent by the user in Figure 12, where the first 2 registers are the data set by the user and the last 2, the data received after decoding.

Channel 2 allows us to know when the data is starting. Its goal is to indicate the start of each preamble. Channel 3 is set to 1 when there is data decoded and there has been an

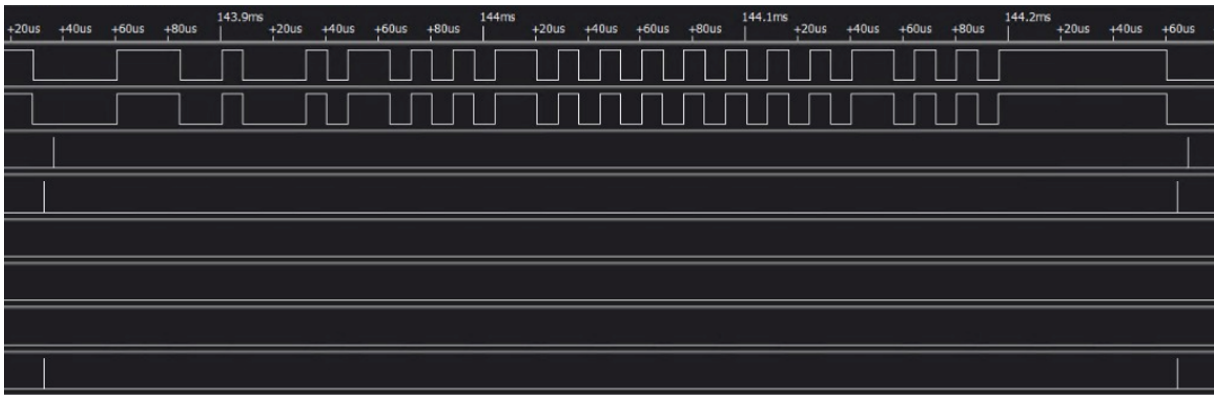

Figure 11: Analysis Hamming communication with a Logic Analyzer

(0x00000018 +) 0x0000	LE	hex	000000FF	GET	SET	Update on Poll
(0x00000018 +) 0x0004	LE	hex	55555555	GET	SET	Update on Poll
(0x00000018 +) 0x0008	LE	hex	00000000	GET	SET	Update on Poll
(0x00000018 +) 0x000C	LE	hex	000000FF	GET	SET	Update on Poll
(0x00000018 +) 0x0010	LE	hex	55555555	GET	SET	Update on Poll

Figure 12: GUI for Hamming codification data transmission

(0x00005170 +) 0x0000	LE	dec	239	GET	SET	Update on Poll
(0x00005170 +) 0x0004	LE	hex	53DE7000	GET	SET	Update on Poll
(0x00005170 +) 0x0008	LE	hex	0CE80101	GET	SET	Update on Poll
(0x00005170 +) 0x000C	LE	hex	00000021	GET	SET	Update on Poll

Figure 13: History buffer records for Hamming codification data transmission

error (either single or double). Thus, in these cases, it allows to confirm the hard coded error in the first bit of the payload.

Finally, channel 7 indicates the end of transmission; it raises a flag once the data has been decoded, and is ready to be sent to the next part of the program.

As we can see in the logic analyser, following the algorithm described in section 3, there is an error in the first bit of payload data transmitted (position 3 in the code-word data, after the preamble bits). Channel 3 also reveals the error, but the register in the GUI corresponding to the received data is storing the correct data. Therefore, this experiment shows how Hamming codification allows the perfect correction of single bit flip.

On the other hand, even if channel 3 is able to show that an error has been found in the data received, the history buffer also allows us to discover the type of error detected. In this case, figure 13 shows us that the type of error stored is 0101 (looking at the bottom 16 bits of the third register of the history buffer), meaning a single error has occurred.

In overall, this section has demonstrated that a new communication's protocol can be implemented in the BIS elements and overcome the limitations presented by the current modulation. In the next section, a comparison of both protocols is done analysing the results shown by this experiment.

8 Results

As shown in section 7, Hamming codification for safety critical communications is a feasible solution to overcome the limitations presented in section 3. In this section, a comparison between the current and the new communication protocols is done to ensure or propose the most robust codification.

8.1 Hamming codification vs Manchester modulation

After deploying the Hamming code solution in a realistic environment, a comparison between Manchester modulation with a single bit parity check and the new Hamming protocol is done. This comparison is based on the self-loop simulation used in section 7.

The use of simulations for the comparison of both methods is done due to the similarities in a real scenario (since the modules are just an adaptation of the ones used for simulations).

Also, the use of simulations allows us to abstract the communications protocol from the overall project, considering only the transmission and reception of the data instead of possible scenarios unrelated to the communication of the BIS elements.

8.1.1 Timing and robustness comparison

The comparison is done considering two possible scenarios: one where the data is clean and another one where a bit flip occurs.

The first scenario is simulated and analysed in order to understand how both communications protocols behave in terms of speed. Figure 14 shows the overall time spent (transmission + reception) for different bytes of data payload:

As we can see, the difference in time when using Hamming as the principal codification, represents a decrease of 50% of the time needed for Manchester modulation since the very first byte of transmission. Indeed, we can see how Hamming is not only faster but more efficient when the number of bytes increments. The reason resides in its algorithm, designed for the gain of efficiency when the data payload increases.

On the other hand, Figure 15 shows the behavior of a Hamming codification compared to the current communications BIS system. It is clearly demonstrated that Manchester modulation with a single even parity check is very inefficient when a single bit flip occurs.

These two comparisons leads us to the conclusion that Hamming codification is not only faster, but also more efficient and robust than a Manchester modulation with a single even parity check bit.In addition, Hamming codification also provides double error detection.

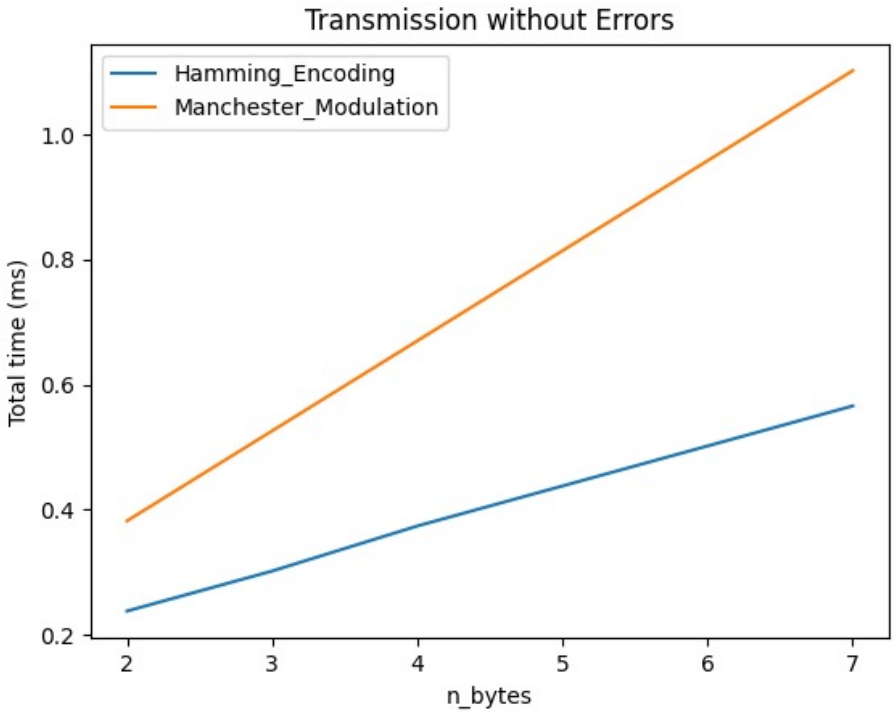

Figure 14: Manchester modulation vs Hamming codification with 0 errors data

In conclusion, even though there are not timing restrictions in the BIS (as explained in section 1, with 100us of operation time, there is no need of a faster protocol), Hamming codification is a solution to big data transmission inefficiency, double error detection and real-time single bit flip correction problems.

8.1.2 Resources comparison

One of the advantages of Manchester modulation with a single even parity check is the resources consumed by the protocol. As Todd at al. [6] specify, one of the requirements for a safety critical communications protocol is the speed and simplicity.

Even though we have seen that Manchester modulation has no timing limitations for the BIS elements, nowadays, Hamming codification is presented as an update of the communication protocol for faster communications. This leaves a margin for technology improvements without affecting timing requirements.

On the other hand, a comparison of the simplicity between both protocols is done to ensure that overcoming the limitations of the current communications protocol is not affecting the requirements of the system.

The resources used when synthesising each design was the primary metric when comparing the two protocols; a simpler design oftentimes uses fewer resources, improving efficiency. Therefore, a design that consumes fewer resources at synthesis, results in a more efficient

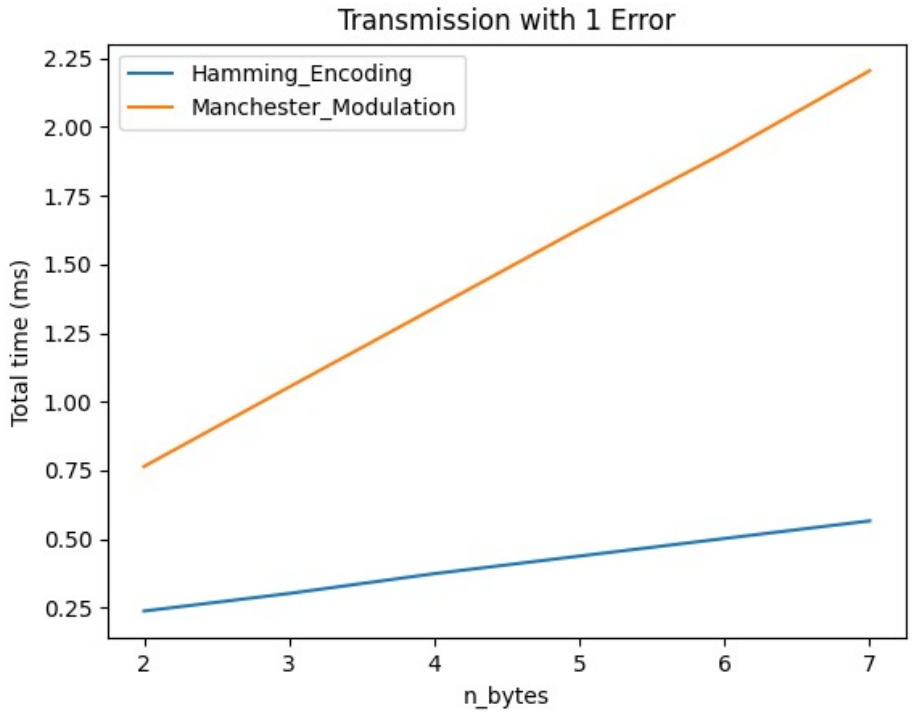

Figure 15: Manchester modulation vs Hamming codification for single bit flip

design overall

Thus, the resources used as a metric for comparison are: (LUT) to understand how many operations the protocol has, Input Output pins (IOs) and the consumption of RAM, to understand how many signals we need and how much processes will be stored in short term.

Table 4 shows the comparison of the resources used by both codifications for a transmission of 5 bytes of data (the standard transmission in the BIS).

Table 4: Resources comparison for Manchester and Hamming protocols

Protocol used	**LUTs**	**IOs**	**RAM**
Manchester modulation	0.71%	40.53%	0%
Hamming codification	1.72%	40.53%	0%

As we can see, neither design require the usage of RAM elements. This is because is because we need to calculate the parity etc. in a case by case basis, using only the information contained within the payload. This payload is always transmitted, without the need to store any information about it afterwards. Taking a look in the algorithm described in section 5, we could think that the difference between Manchester modulation and Hamming when using LUTs will be significant. The not significant difference is due to the use

of variables and processes instead, especially for the value calculus of the redundant even parity check bits.

On the same subject, the use of processes and variables allows us to decrease the number of IOs and avoids the use of RAM for Hamming codifications, making it equally effective as the Manchester modulation.

In overall, the overcoming of the current limitations set by the Manchester modulation is efficiently done by the Hamming codification. This allows the possibility of future system technology updates without communications restrictions.

9 Budget

The project is based on a protocol framework for safety critical communications. The software characteristics of the solution, does not have any production related costs. Hence, the costs associated to this project are those related to the hardware and information required for performing the research.

The project is based on the BIS system characteristics. The hardware used for the deployment of the project is not considered in the budget, as it was already existing before the start of the project.

For the ID framework design and deployment we only need a single engineer to carry out the project. Since the author is considered a junior engineer, the costs of the author's salary are reduced to 13€/h.

The Hamming codification framework is implemented inside an FPGA and simulated in a computer. The costs of a computer have to be computed (1000€). In order to compute them, we will consider its depreciation. The depreciation period of the computer is 5 years with a residential value of 200€. Then, the depreciation value of the computer for 4 months is 67€.

Finally, Table 5 shows the total budged required for the project's deployment.

Table 5: Project budget

Details	Count	Unitary	Amount
Engineer salary	300h	13€ / h	3900€
Computer depreciation	4 months	200€ / year	67€
Internet Connection	4 months	35€ / m	140€
Total			**4.107 €**

10 Sustainability

The aim of this project is to ensure a robust system for safety critical communications without creating a bad impact in the sustainability. In order to minimize the environ-

mental impact, a study of the project's social, economic and security impact has been done.

10.1 Social Impact

Currently there is a great diversity of applications for the LHC accelerator (as explained in the experiments presented in section 1).

Having a robust communication channel allows the correct development of these experiments and their applications. Some of the applications extracted from the LHC experiments focus on: medical services [18], information and technology applications [19], mechanical and construction updates, etc. The deployment of this framework, adds efficiency to those applications since it allows for technology applications a better understanding of the data received and for the use of magnets the ability of error correction in noisy environments.

10.2 Economic Impact

In addition to the previous applications for accelerators, the deployment of an efficient communications system allows us to optimise the protection of CERN accelerators. This avoids the loss of important information that can lead the beam to a false dump, having a huge economic impact if the protocol is not transmitting the correct information [6].

Hence, our project represents a new opportunity in the upgrade of accelerators technology, guaranteeing its safety critical communications.

10.3 Security Impact

Finally, our project has a fundamental role in the security impact. First, the correct collection of data is crucial for the future of the beam, since the BIS is the heart of the MPS for the LHC.

Second, an advantage for the society is improving the security in safety critical systems. LHC safety critical systems sets a new concept methodology for other similar systems. Thus, many applications based on safety critical communications and safety critical systems were and are currently also used after its methodology is tested and explained in [20], allowing security transfer knowledge to society.

Finally, the security of the BIS channel ensures a good monitoring in different parts of the accelerator. This monitoring helps understanding and preventing beam and energetic disasters. By having a secure channel, the beams can efficiently be used for many different applications.

In conclusion, after evaluating the different environment impacts in the project, sustainability is ensured with the security of safety critical communications.

11 Conclusions

Safety critical communications are the core of the MPS for the LHC accelerator. The update and improvement of these communications must be done in line with the technological changes achieved in the accelerator itself (for instance the increment of energy per beam).

Considering the future of the experiments and the requirements of the BIS in terms of reliability and robustness [16], the development of a new communications protocol is designed and validated.

Different scenarios and approaches (single bit flip, double error detection, etc.) are presented and a new challenge is faced: the same module's structure has to be follow for Hamming codification. This challenge is set for maintainability purposes (to distribute the different elements that constitutes the overall BIS project).

Many contributions are done with accurate results such as the ability to correct in real time single bit flips. This avoids the re-transmission of the frame and thus the loss of the entire payload for just a single bit error. In addition, a double error detection is offered, representing an improvement of the protocol robustness.

Another big contribution is the insurance of a valid frequency transmission rate of 62,5kHz, also due to the correction in real time of bit flips in the payload.

Finally, the limitation of coding only byte by byte in the entire payload transmitted to ensure a reliable communication is overcome with Hamming codification, since the protocol increases efficiency with the increment of data transmitted.

The use of simulations and the deployment of a real case scenario, have proven the ability a Hamming codification has to ensure a fast, reliable, and robust safety critical communication. This offers the possibility to use this codification in the BIS and set a precedence and update for the communications in the MPS.

As a final conclusion, with the design of the Hamming codification for safety critical communications and its deployment in a real case scenario, we found a solution to the limitations found in the current communications system without affecting the resources nor the complexity of the system.

12 Limitations and Future Work

The aim of this section is to present the limitations and the next steps of the project.

After targeting the problem, the project is only focused on BIS communications. Even though this system represents the heart of the MPS, other systems with different characteristics may consider the implementation of a new codification. Thus, a new line of research can be opened for the implementation of Hamming codification in different critical systems.

On the other hand, delving deeper into the upgrade of the BIS system, Secondo et al.[21] presents the introduction of Small Form-factor Pluggable transceivers SFPs for one of

the permits set by the system. This new update consists in the transmission of safety critical information using fiber optic. Considering the characteristics of the fiber optic, we found the following limitation: without a modulation nor a transition between the different bits transmitted, the use of Hamming codification for fiber optic may lead to the loss of communication. This is because fibre works best with a self-clocking protocol. Thus, Hamming codification is limited for fiber optic transmissions.

Finally, after ensuring good results based on Hamming theorems, the deployment of different techniques for safety critical communications can be done. A survey on these techniques may help us to find combined methodologies for a more robust, faster, and efficient communication between critical elements in the LHC.

References

[1] ATLAS Collaboration, G Aad, E Abat, J Abdallah, AA Abdelalim, A Abdesselam, O Abdinov, BA Abi, M Abolins, H Abramowicz, et al. The atlas experiment at the cern large hadron collider, 2008.

[2] Marcela Carena and Howard E Haber. Higgs boson theory and phenomenology. *Progress in Particle and Nuclear Physics*, 50(1):63–152, 2003.

[3] Serguei Chatrchyan, Gevorg Hmayakyan, V Khachatryan, CMS Collaboration, et al. The cms experiment at the cern lhc. *Journal of instrumentation*, 3(8):S08004, 2008.

[4] Kenneth Aamodt, A Abrahantes Quintana, R Achenbach, S Acounis, D Adamová, C Adler, M Aggarwal, F Agnese, G Aglieri Rinella, Z Ahammed, et al. The alice experiment at the cern lhc. *Journal of Instrumentation*, 3(08):S08002, 2008.

[5] A Augusto Alves Jr, LM Andrade Filho, AF Barbosa, I Bediaga, G Cernicchiaro, G Guerrer, HP Lima Jr, AA Machado, J Magnin, F Marujo, et al. The lhcb detector at the lhc. *Journal of instrumentation*, 3(08):S08005, 2008.

[6] Benjamin Todd. *A beam interlock system for CERN high energy accelerators*. PhD thesis, Brunel U., 2006.

[7] Zidong Wang, Licheng Wang, Shuai Liu, and Guoliang Wei. Encoding-decoding-based control and filtering of networked systems: insights, developments and opportunities. *IEEE/CAA Journal of Automatica Sinica*, 5(1):3–18, 2017.

[8] Saleh Faruque. Introduction to channel coding. In *Free Space Laser Communication with Ambient Light Compensation*, pages 1–15. Springer, 2021.

[9] George C Clark Jr and J Bibb Cain. *Error-correction coding for digital communications*. Springer Science & Business Media, 2013.

[10] Sanjeev Kumar and Ragini Gupta. Performance comparison of different forward error correction coding techniques for wireless communication systems. 2011.

[11] Jatinder Singh and Jaget Singh. A comparative study of error detection and correction coding techniques. In *2012 Second International Conference on Advanced Computing & Communication Technologies*, pages 187–189. IEEE, 2012.

[12] Ali Calhan, Celal Ceken, and Ismail Erturk. Comparative performance analysis of forward error correction techniques used in wireless communications. In *2007 Third International Conference on Wireless and Mobile Communications (ICWMC'07)*, pages 63–63. IEEE, 2007.

[13] Manika Pandey and Vimal Kant Pandey. Comparative performance analysis of block and convolution codes. *International Journal of Computer Applications*, 119(24), 2015.

[14] Caleb Hillier and Vipin Balyan. Error detection and correction on-board nanosatellites using hamming codes. *Journal of Electrical and Computer Engineering*, 2019, 2019.

[15] Rudolf Ahlswede and Gyula OH Katona. Contributions to the geometry of hamming spaces. *Discret. Math.*, 17(1):1–22, 1977.

[16] R Johnson, C Martin, T Podzorny, I Romera, R Secondo, J Uythoven, et al. The consolidation of the cern beam interlock system. 2021.

[17] Declan O'Loughlin, Aedan Coffey, Frank Callaly, Darren Lyons, and Fearghal Morgan. Xilinx vivado high level synthesis: Case studies. 2014.

[18] Ram Gopal Sharma. *Superconductivity: Basics and applications to magnets*, volume 214. Springer Nature, 2021.

[19] Dan Guest, Kyle Cranmer, and Daniel Whiteson. Deep learning and its application to lhc physics. *arXiv preprint arXiv:1806.11484*, 2018.

[20] Francesco Valentini, Timo Hakulinen, Louis Hammouti, Tomasz Ladzinski, Pierre Ninin, et al. Formal methodology for safety-critical systems engineering at cern. In *Proceedings of the 14th International Conference on Accelerator & Large Experimental Physics Control Systems*, pages 918–921, 2013.

[21] Raffaello Secondo, Marc-Antoine Galilée, Jean-Christophe Garnier, Christophe Martin, Iván Romera, Andrzej Piotr Siemko, Jan Uythoven, et al. Evaluation of an sfp based test loop for a future upgrade of the optical transmission for cern's beam interlock system. 2019.

Appendices

A Work Plan

The work plan represents the different processes of the project. The project is divided in five work packages. Each of them has a specific purpose related to the methodology of the project.

- **Theoretical background study (WP1):** The Theoretical background work package consists of a deep investigation of the current system where the project takes place, in order to have a better understanding of the field. Here there are 3 tasks:
  - Study of the LHC and its systems.
  - Understanding of the MPS.
  - Analysis of the BIS elements.
- **Research of different communications protocols (WP2):** Elaborate a list of possible codification alternatives based on the previous background study. Here there are 5 tasks: Understanding of the current Protocol (Manchester modulation) (T1), Find the limitations of this protocol (T2), find alternatives based on the limitations to overcome (T3), understand the complexity and limitations of each alternative (T4) and the study of the final selected alternative (T5).
- **Design of the Hamming encoder/decoder(WP3):** Elaborate a Hamming communications protocol efficiently. Simulate the protocol with the same specifications as the Manchester modulation.
- **Comparison and understanding the new protocol (WP4):** A comparison has to be done an the practical experiment (use case scenario) has to be set up.
- **Testing and visualization of the final result (WP5):** Test the Hamming encoder in a real case scenario.

The GANTT diagram for all the work packages is illustrated in Figure 16.

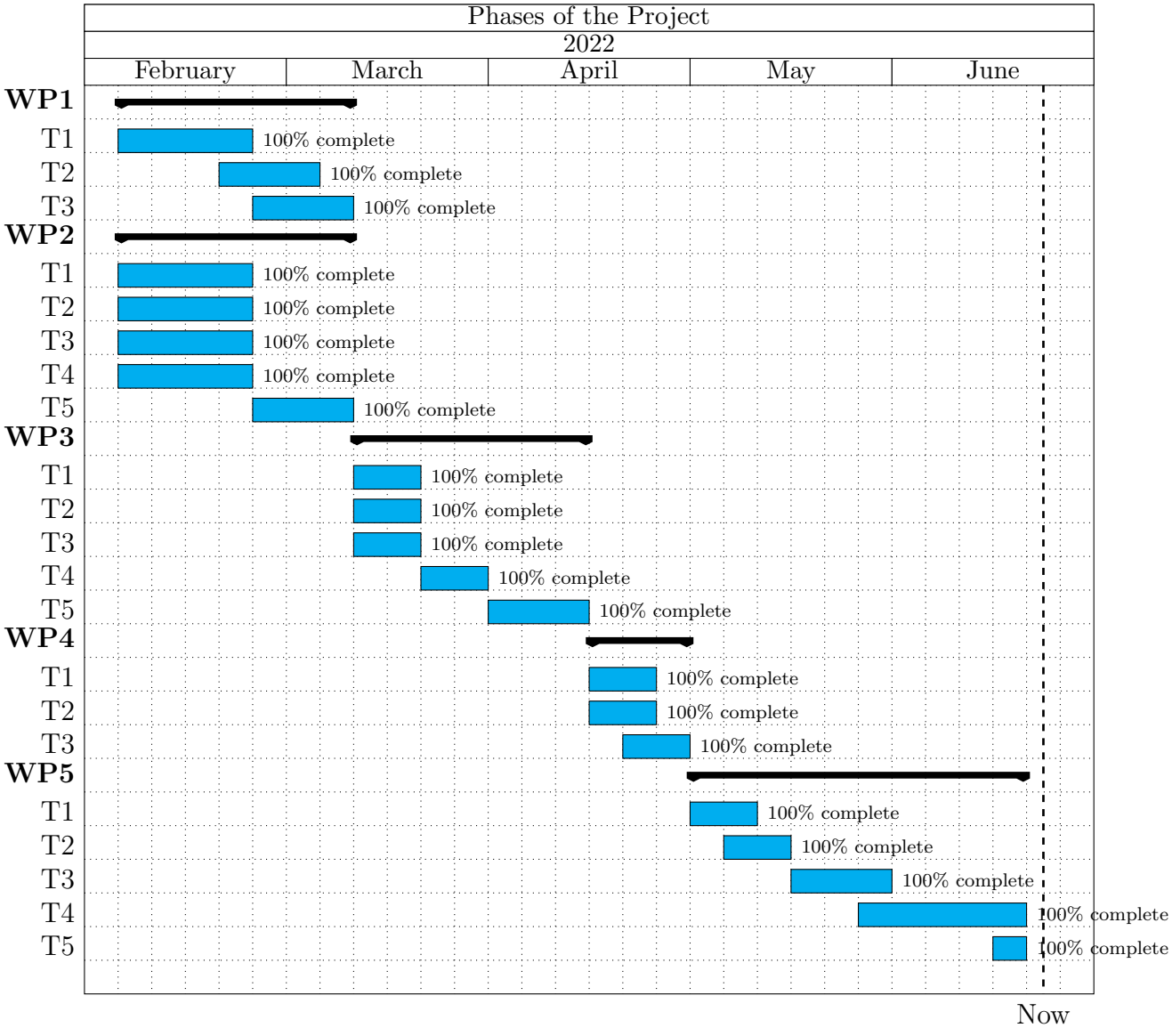

Figure 16: Gantt diagram of the project