Proposal for a specific (128,120) extended inner Hamming Code with lower power and lower latency soft Chase decoding than textbook codes

September 23, 2022

Will Bliss will.bliss@broadcom.com
Maged F. Barsoum maged.barsoum@broadcom.com
German Feyh german.feyh@broadcom.com

Overview of Previously Proposed Inner Concatenated Hamming Codes

- A concatenated coding system for 200G/ lambda optical segments remains the only proposed method to achieve significant net coding gain with reasonably low latency and power
  - Outer code remains KP4, leaving the IEEE 802.3 format unmolested
  - Interleaving of KP4 codewords has been shown necessary for maximum coding gain
  - Note that interleaving of Hamming codewords hasn't been discussed, but a very simple adequate solution is presented
- Inner code (128,120) extended Hamming code has been proposed
  - E.g, patra_3df_01_2207, bliss_3df_01c_220517
  - Rate 15/16 ➔ Baud rate = 113.333.. Gbaud/sec (not a multiple of the 156.25 MHz common Ethernet Xtal)
  - Proposed by individuals tightly coupled to IC implementation
- Inner code (144,136) Hamming code has been proposed
  - E.g., he_3df_01a_220308,
  - Rate 17/18 ➔ Baud rate = 112.5 Gbaud/sec = 720* 156.25 MHz Ethernet common Xtal
    - Avoids Frac-N PLLs, generally allows lower jitter PLLs
  - Generally preferred by the Ethernet community

A Compromise Inner Concatenated Code Proposal

- Reuse the (128,120) rate 15/16 code hardware by shortening that code, WHILE simultaneously
- Achieving the higher rate of 17/18, preserving the Ethernet common compatibility
  - Which sounds impossible, but it is possible because;
- The Inner Code only needs to protect PAM-4 symbols (Bauds), not binary bits
  - Every given noisy Gray-coded PAM-4 received symbol can only have one low reliability bit (either the MSB or the LSB, but never both bits)
    - Easy to see for a PAM-4 slicer. IF a low reliability bit, the probability the 'other bit' is actually in error is so rare as to not impact system performance
  - Define the data payload of the shortened code as the XOR(MSB,LSB) of each of the 136/2 = 68 data PAM-4 symbols in a *(144,136) format*
    - IF a soft correction points to one of these data Bauds, only consider flipping the low reliability bit
  - The 8 parity bits are transmitted normally as 4 PAM-4 symbols
  - The net *binary* code is thus (76,68), which can be implemented as a shortened version of an extended Hamming (128,120) code.
  - The hardware for implementing the shortened code is the ~same algebra as for the full code, and there are only 76/128 = 59.3% as many locations to consider correcting

Compromise Inner Code of Rate = 17/18

136 bits payload | 8 bits parity

68 PAM-4 symbols payload | 4 PAM-4 symbols parity

Physically on the line. Rate is 136/144 = 17/18

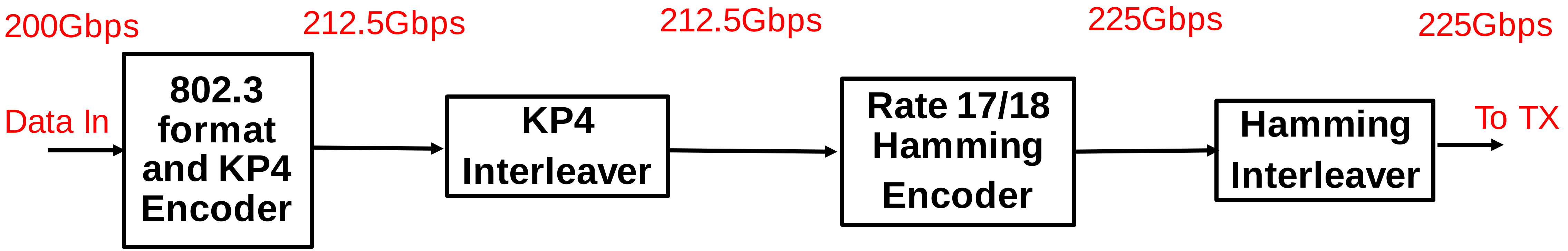
XOR(MSB,LSB) reduces the code payload from 68 Bauds to 68 bits

The code format is binary (76,68), which can be a shortened extended Hamming (128,120) code

68 bits ‘code’ payload | 8 bits parity

Decoding the shortened code points to the PAM-4 symbols to ‘correct’, which is unambiguous and essentially lossless compared to pointing to ‘bits’

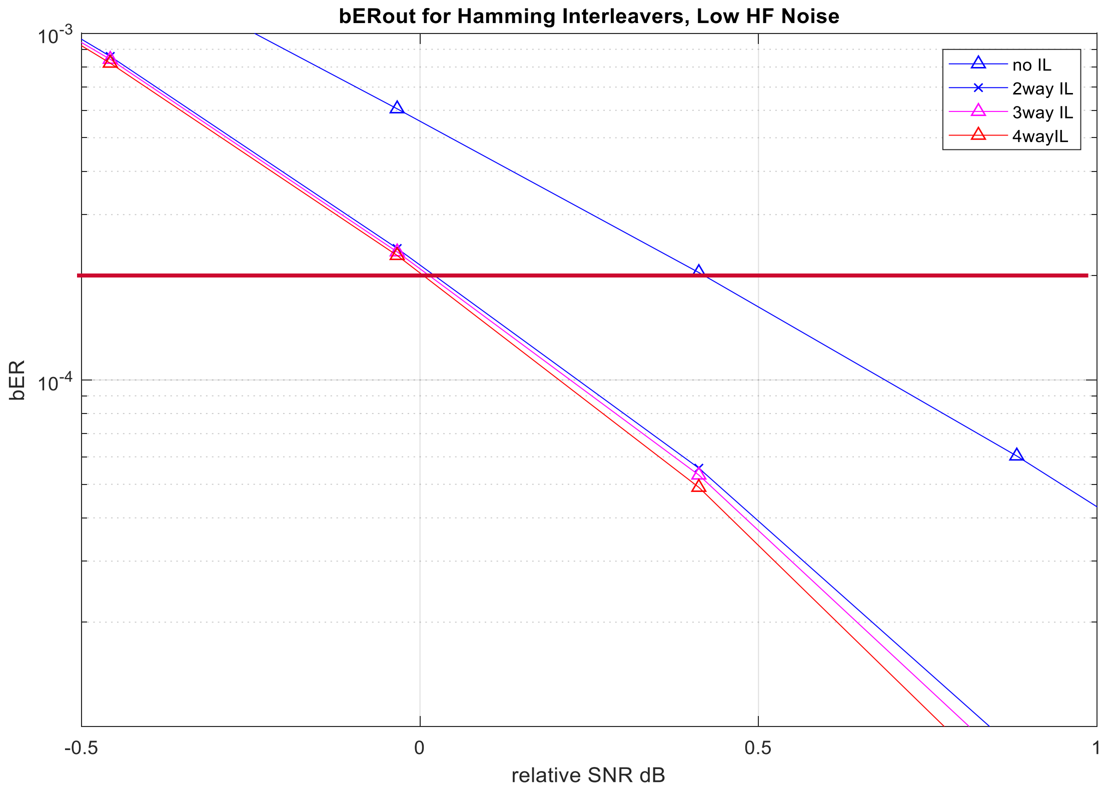
Block Diagram of the Proposed System TX / Encoding



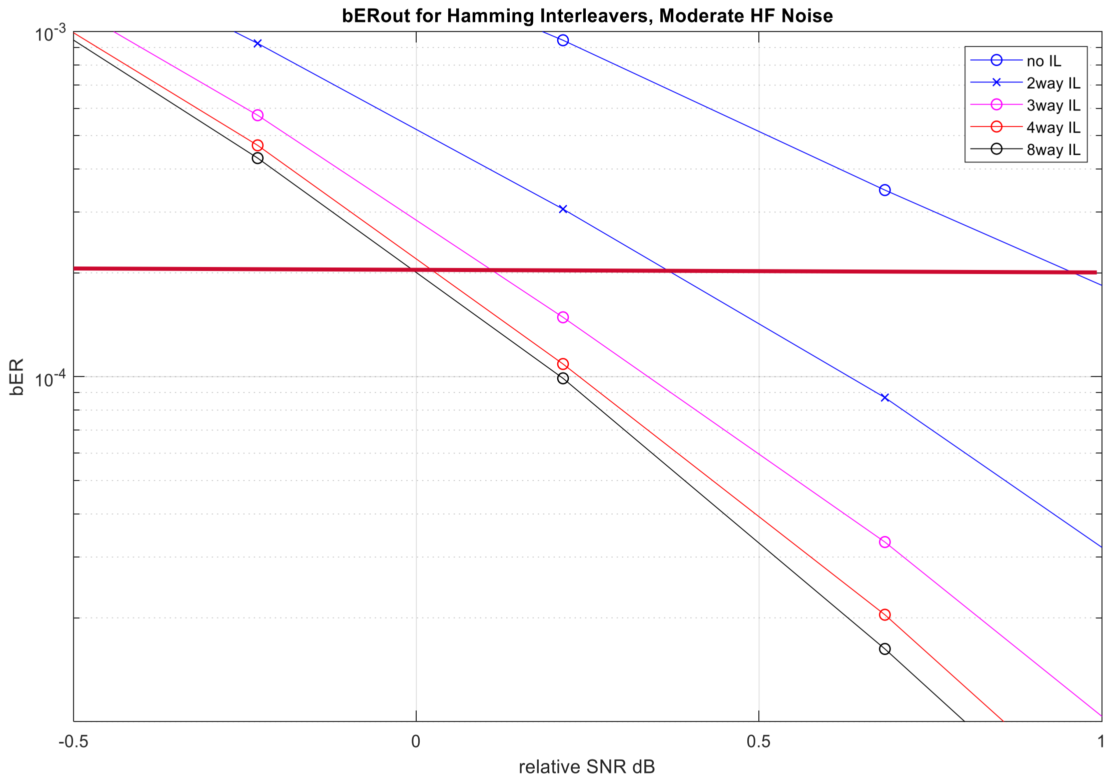
- Baud rate is ‘802.3 KP4 rate’ * 144/136 = 106.25G * 18/17 = 112.5 Gbaud/sec
- Proposal is to use a shortened (128,120) extended Hamming code for PAM-4 to effect rate 17/18
- KP4 interleavers have been described previously for rate 15/16. Increase from 12-way to 14-way for the 17/18 rate
- A Hamming interleaver based on Baud (PAM-4) units is needed to achieve high coding gain IF the SNR(f) is low at high frequencies, which creates some error patterns of several Bauds in length
  - Because Hamming codewords are very short compared to KP4 words, and because we only need to spread out adjacent PAM-4 symbols (not RS symbols), the cost and latency for even 8-way Hamming Baud interleaving is low

Soft Hamming Performance vs 'Hamming Interleaver Ways'

w/ Minimal SNR degradation at high freq.



w/ Moderate SNR degradation at high freq.



- A Hamming Codeword interleaver based on Baud (PAM-4) units is needed
  - 2-way may be enough for 'good channels' with only moderate SNR(f) degradation at high frequency
  - >= 4-way is needed for moderate SNR(f) degradation at high frequency
  - Because Hamming codewords are very short compared to KP4 words, and because we only need to spread out adjacent PAM-4 symbols (not RS symbols), the cost and latency for even 8-way Hamming Baud interleaving is low, and provides protection against 'even worse' SNR(f) at high frequency

Chase type Soft Hamming Decoding Algorithm

- Let N= $\binom{q}{1} + \binom{q}{2} + \ldots + \binom{q}{w}$
  - All combinations of the q lowest reliabilities, taken at up to a maximum of w at a time
- IF syndrome==<0 0 0 0 0 0 0 0>, THEN done; Else continue
- For k=1:N where N= is the number of 'test patterns'
  - Flip the w(k) <=w bits in this test pattern
  - *Run Hard Hamming Correction* for this test pattern
    - *Look Up the Reliabilty for this 'hard correction bit'*
  - Calculate the net reliability for this case of w(k)+1 bit flips, and save
- END
- Output the corrected codeword with the highest Net Reliability
    - Strip off the Hamming Parity bits and deliver to data sink

This operation motivates the proposed Code to reduced complexity

The Design Problem

- Find / Create a systematic (128,120) extended Hamming code with a low cost hard Hamming parallel decoder
  - Only matrix G and H descriptions are considered, because high speed Parallel operation is required
  - Code choice is not limited to ‘polynomial type serial’ descriptions
- Certain linear operations are allowed on the parity portion of the generator G matrix
  - Except the *kxk* = 120x120 Identity matrix must remain to be a Systematic code
- Certain linear operations are allowed on the full parity check H matrix
  - A given G matrix can have a huge number of functional H matrices
  - The canonic form where H contains an Identify matrix isn’t required
- A directed search found the following systematic code with a simple map from syndrome to location

Generator Matrix

The proposed 128x120 Systematic encoding G matrix is

$G=[P^T_{8x120} \,;\, I_{120x120}]$,

Denote the data payload = u, and the encoded message = m

$m_{128x1} = G_{128x120} * u_{120x1}$

where the 8x120 matrix $P^T =$

$$\begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{bmatrix}$$

To complete the total definition of the Hamming encoder, we further propose that the parallel data payload u_{120x1} is received 'bottom first' and 'LSB first', and similarly message m_{128x1} is output 'bottom first' and 'LSB first'

This completes the definition of the Hamming encoder, which is usually where standards stop, leaving the receiver / decoder to the implementer

Parity Check H Matrix

The canonic form Parity Check H matrix is

$H_{8x128}=[I_{8x8}\ ,\ (P^T)\ _{8x120}],$

Denote the syndrome = s, and the noisy message = n

$s_{8x1} = H_{8x128}{}^{*}\ n_{128x1}$

The proposed Parity Check Matrix for simplified decoding is $H'_{8x128}=$

$$\begin{bmatrix}
1&1 \\
0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \\
0&0&1&1&0&0&1&1 \\
0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1&0 \\
0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1&0&0&0&1&1&1&1&0&1&1&1 \\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1 \\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1 \\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0&1&1&1&1&1&1&1&0
\end{bmatrix}$$

which gives a ‘different’ syndrome than the canonic form, but a syndrome which makes it simpler to locate an error

Note that users are free to use their own derivations / favorite versions of H. Our favorite is only presented to motivate standardizing the described G matrix, which defines ‘the code’.

Simple Mapping from Syndrome to binary bit location

Denote the syndrome vector s_{8x1} as <S(1), S(2) … S(7),S(8)>

Where S(8) is the top of the s_{8x1} vector and is the extended parity

Denote the binary address of the bit to be corrected by the syndrome as number <i(1), i(2) .. i(7)>

i(1)=S(1) ; i(2)=S(2) ;

i(3)=xor((S(1) & S(2)), S(3)) ;

i(4)=xor((~S(1) & ~S(2) & S(3)), S(4)) ;

i(5)=xor((S(1) & ~S(2) & S(3)), S(5)) ;

i(6)=xor((~S(1) & S(2) & S(3)), S(6)) ;

i(7)=xor((S(1) & S(2) & ~S(3)), S(7)) ;

- Which is implemented using 8 AND and 5 XOR gates
- 11% power reduction to the net soft Chase implementation compared to ROM versions
  - Implementations of Textbook codes with ROMs require throughputs near 100e9 Reads/sec of 128 word ROMs with 7-bit words
    - Depending on choice of Chase algorithm w and q parameters

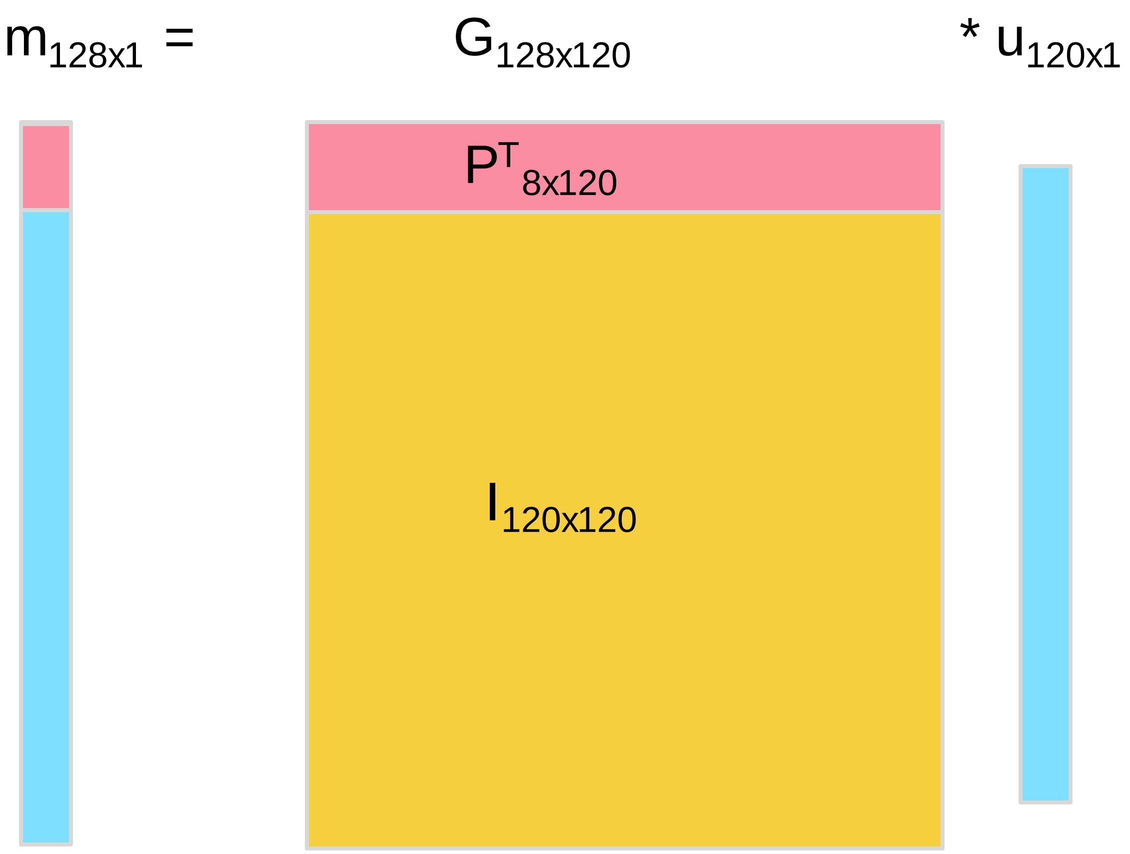
Shortening the (128,120) code to binary (76,68)

- The proposed code, like all systematic codes, is simple to ‘shorten’
- Message bits not used can conceptually be filled with zeros
- Received bits not used likewise can be zeroed out
- Which makes clear the components of G and of H that can be deleted
- The 52 right side columns of G and the bottom 52 rows of G are deleted
- Similarly, the 52 right side columns of H are deleted

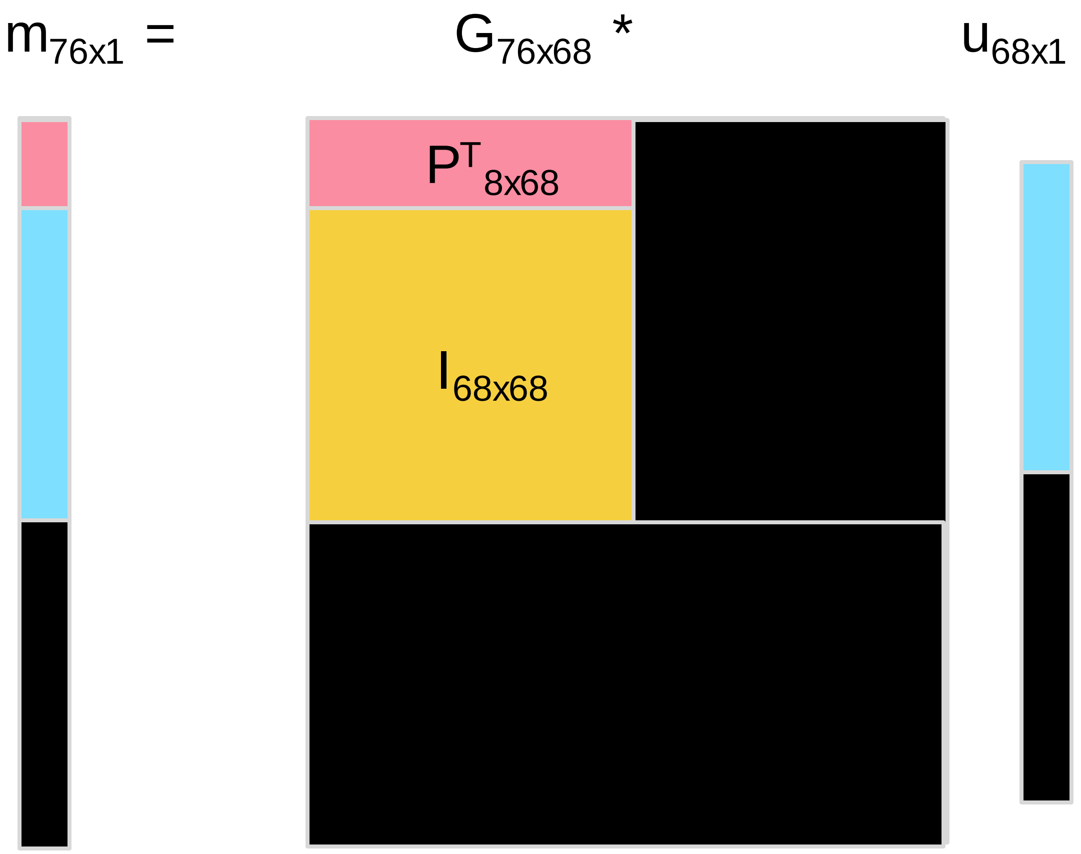
Shortening the (128,120) code to binary (76,68); G

- The 52 right side columns and 52 bottom rows of G are deleted

Full length code

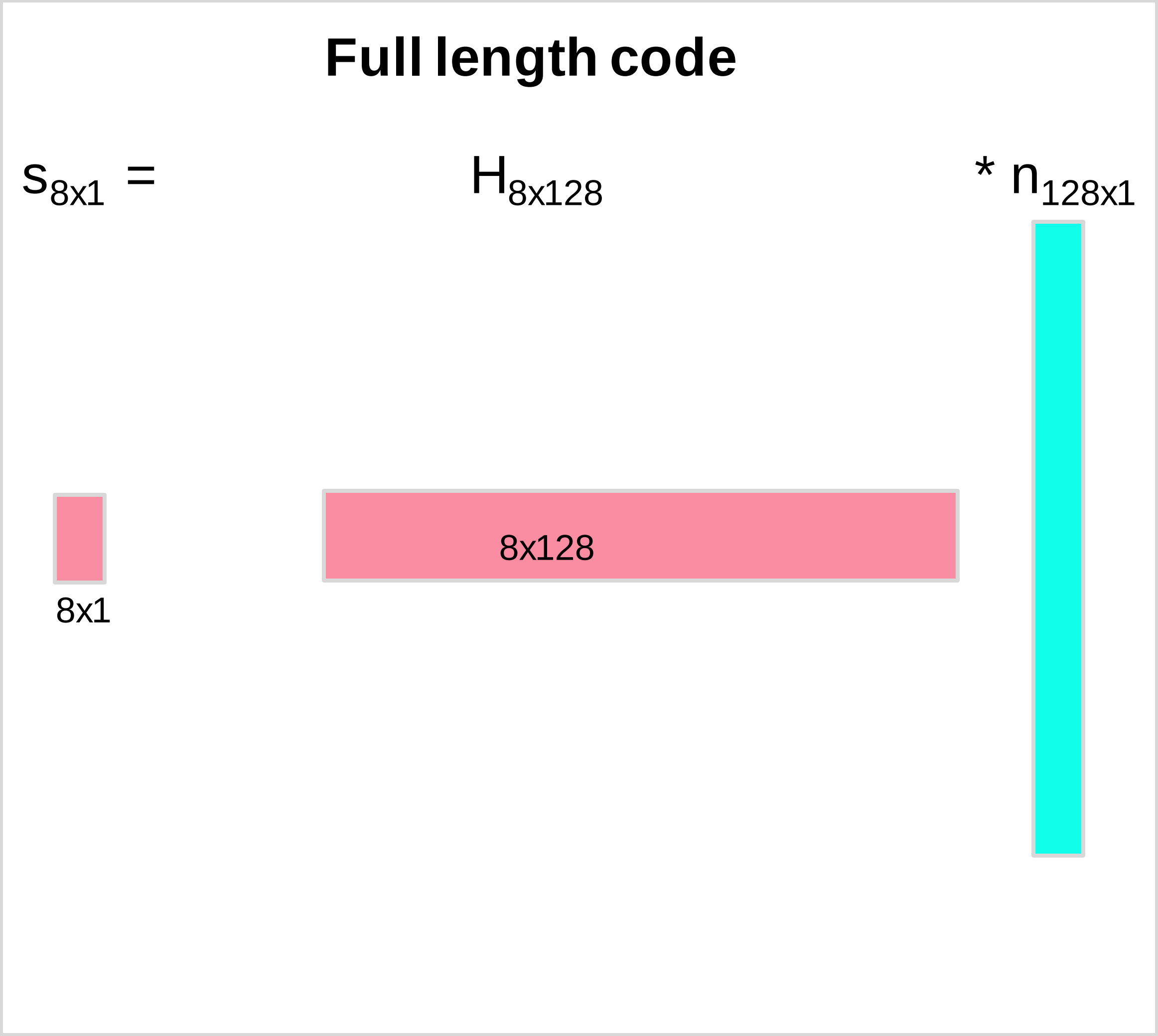


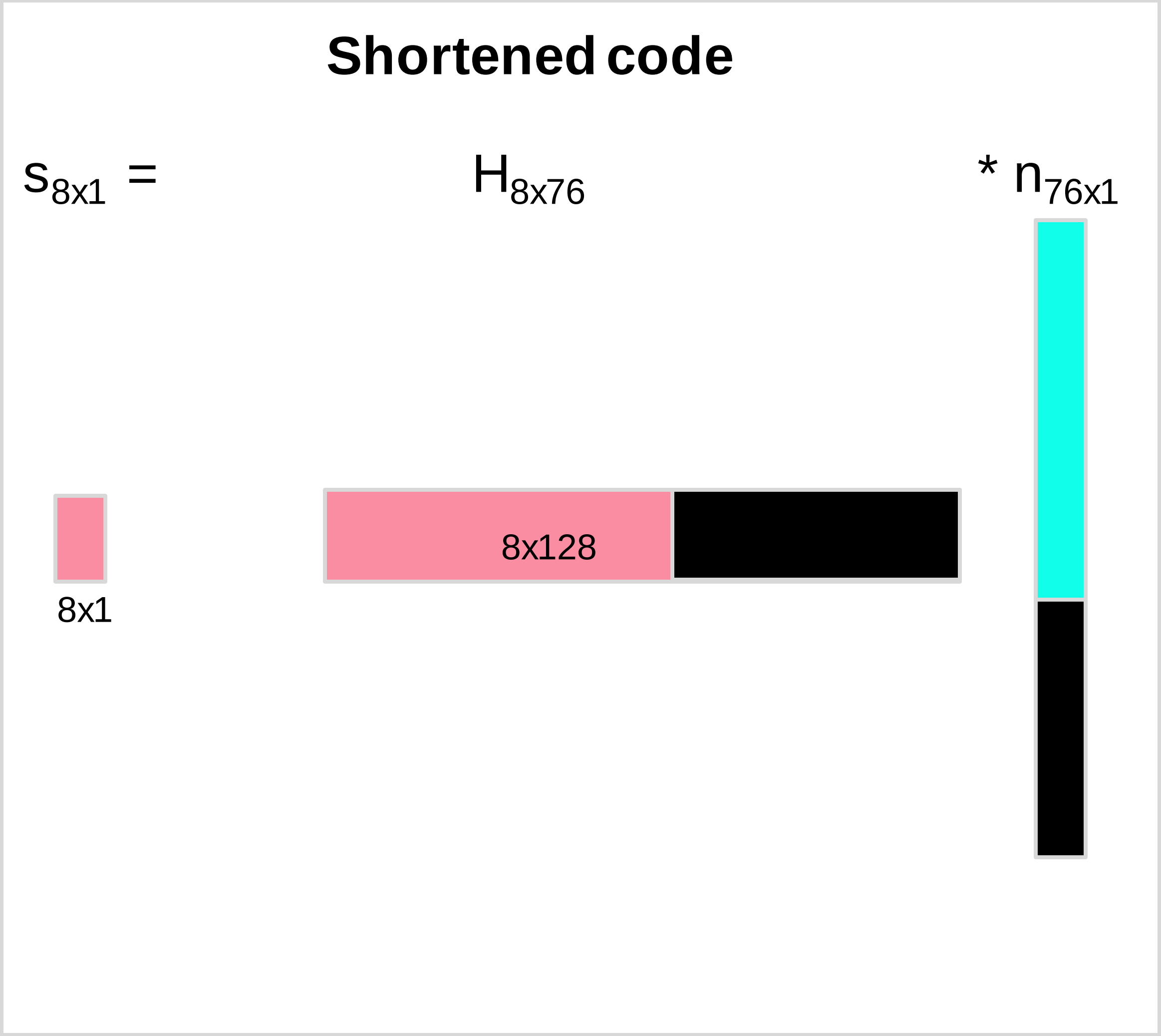
Shortened code



Shortening the (128,120) code to binary (76,68); H

- The 52 right side columns of H are deleted





Summary of Proposal

- A compromise Inner Hamming Coding Method is proposed that
  – Achieves rate 17/18,
    – which most Ethernet users prefer as it can share the common 156.25MHz Xtal reference
  – Allows use of *any* (128,120) binary extended Hamming code with simple shortening
    – Which silicon developers have preferred
- A specific (128,120) extended Hamming code is proposed that
  – Includes a detailed description of syndrome generation and mapping to binary addresses
  – Which is simpler to ‘hard decode’ in a soft Chase algorithm than textbook codes

- THANK YOU