

Update and proposal for DSQ128+ for Coding the Unprotected Bits

Paul Langner

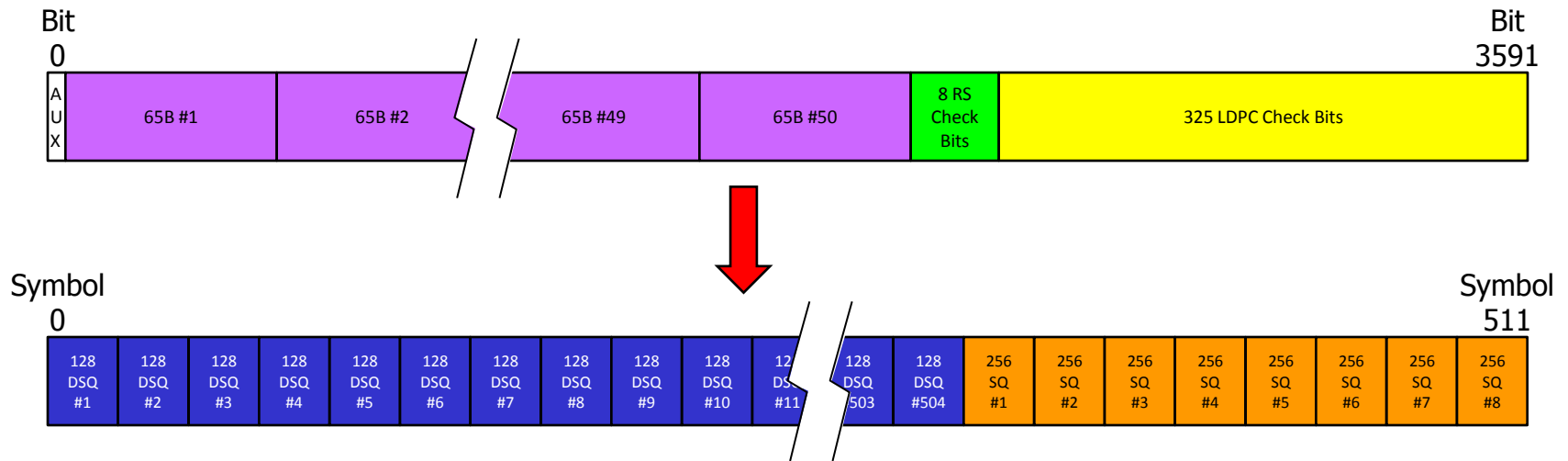
Goal

- Add error correction to the uncoded bits in a standard 128DSQ modulation format
- Target correction is a single error event per frame
- Should re-use as much 128DSQ machinery as possible
- Should re-use as much of the PCS framing machinery as possible

128DSQ + RS256 = 128DSQ+

- 128DSQ contains $512 \times 3 = 1536$ uncoded bits = 192 bytes
 - A good fit for standard GF256 Reed-Solomon (RS256), which can provide error protection on up to 255 byte block sizes
 - For every two check bytes, RS256 can correct one byte (2+ symbols) and detect two errored bytes (5 symbols)
 - Minimum RS256 correction requires 2 bytes = 16 bits
 - Can reuse CRC-8, but need 8 more bits
- Obtain these 8 extra bits by going to full PAM-16 for the last 8 symbols of the 128DSQ frame

128DSQ+ Frame

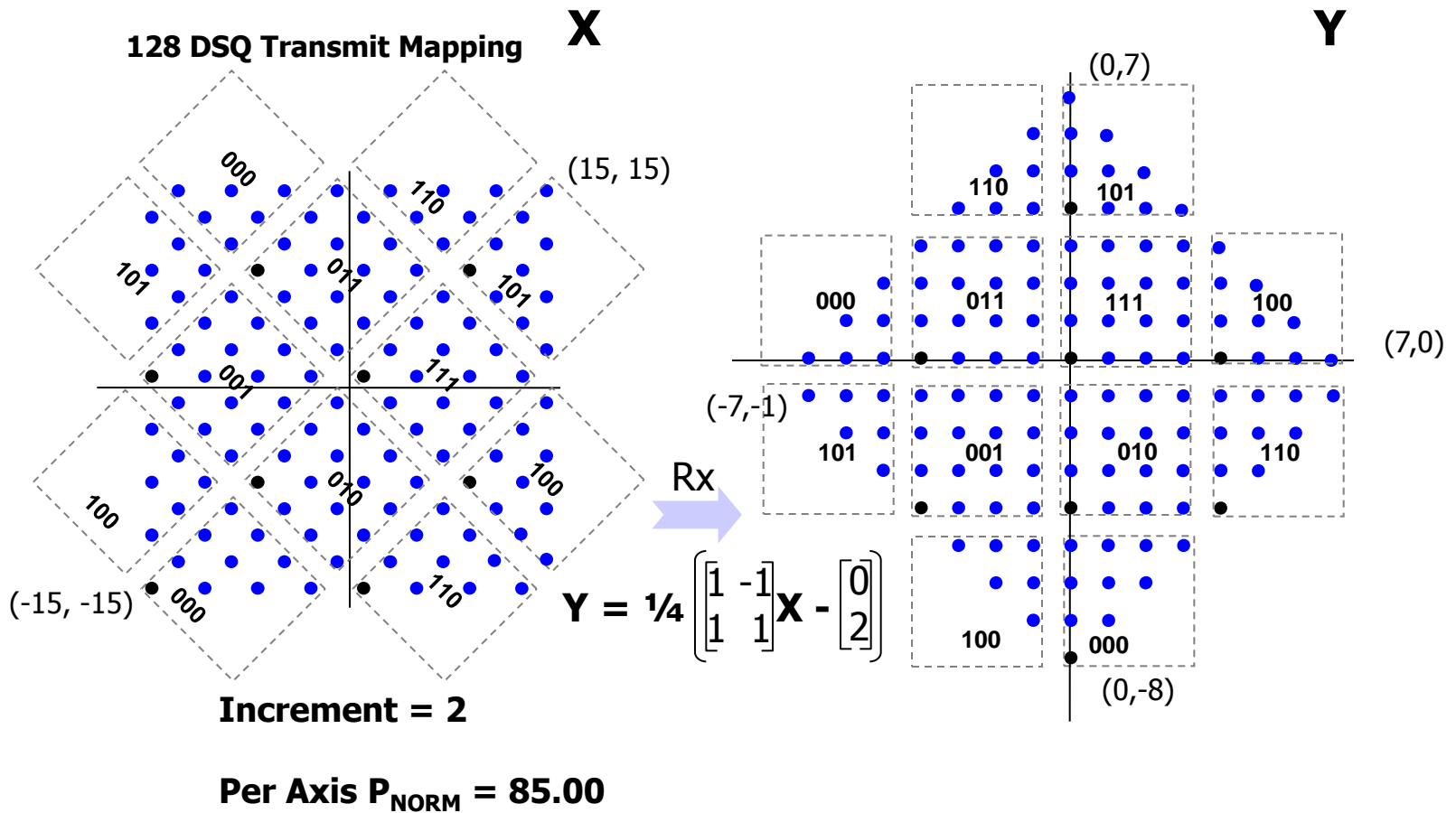


- **Take standard 128DSQ frame, and add 8 extra uncoded bits via transmitting 256SQ on last 8 symbols**
 - 8 CRC-8 bits + 8 new bits -> 16 bits for RS256 over uncoded bits
 - Corrects any 8 bits (2+ back-to-back symbols) in frame and detects double in errors

128DSQ+

- 1. Provides 2+ symbol error correction and 4+ symbol error detection over uncoded bits**
- 2. Incurs a ~ 0.05 dB noise impairment after coding**
- 3. Utilizes standard 128DSQ Rx machinery**
- 4. Unlike CRC-8 corrector, adds legitimate error checking on payload, and superior error correction for a small penalty**
- 5. Data rate stays at 40 Gb/s, symbol rate stays at 3200 MS/s, and frame period stays at 80ns**

128DSQ Mapping



256SQ Mapping

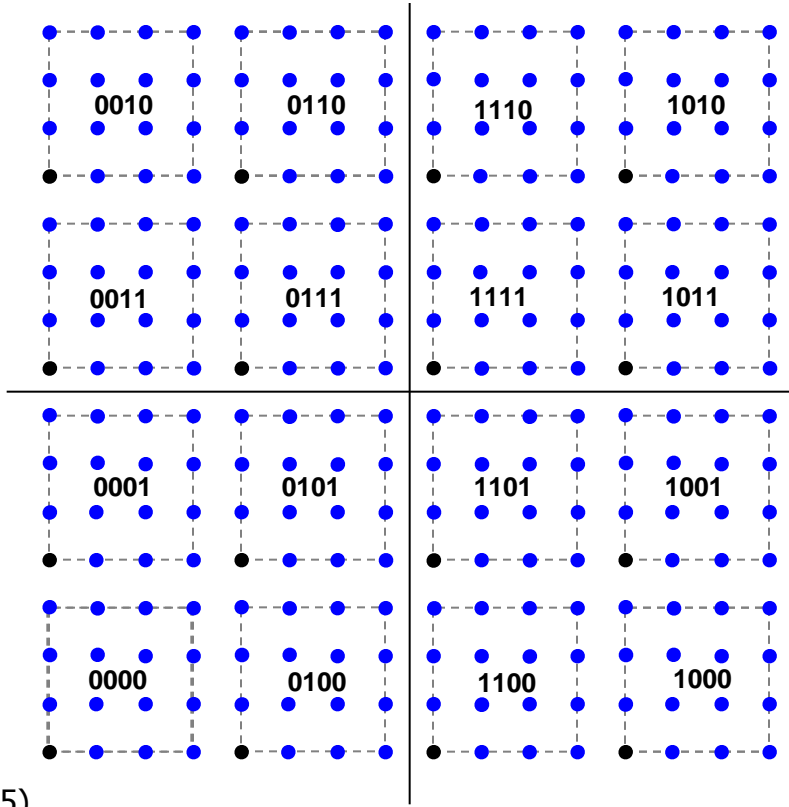
256 SQ Transmit Mapping

X

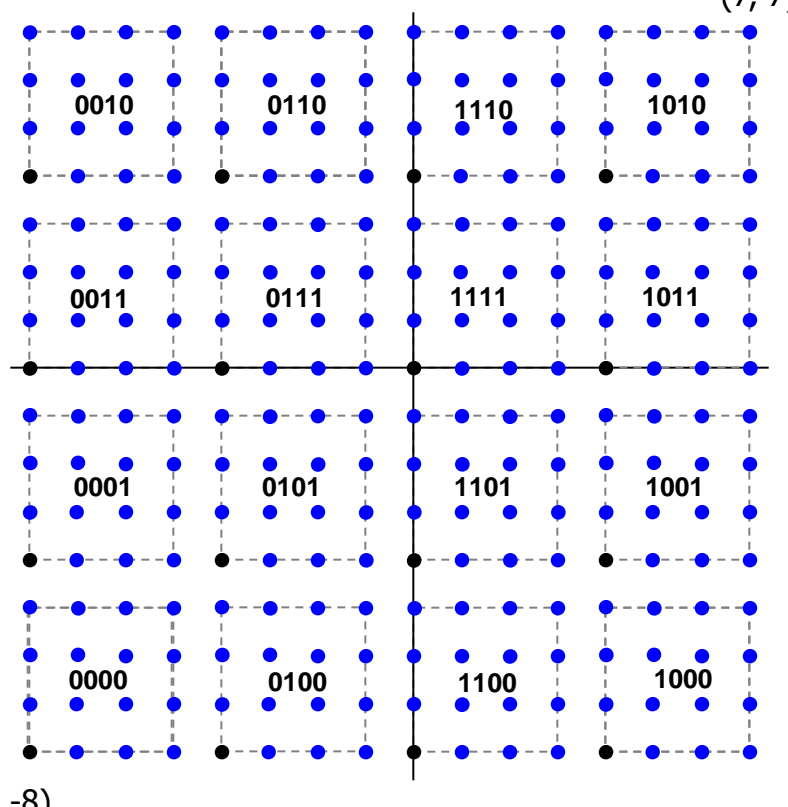
(15, 15)

Y

(7, 7)



Rx



Increment = 2

Per Axis $P_{NORM} = 85.00$

$$Y = \frac{1}{2} \left[X - \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right]$$

Why 256SQ versus PAM-16?

- 1. Maintains concept of uncoded and coded bits**
- 2. Maintains same LDPC LLR calculator as 128DSQ**

Changes From a Standard 128DSQ PCS Required for Implementation

- **Need to implement PAM-16 versus 128DSQ DSP slicer**
- **Need to implement RS (195,193) code**
 - $504 \times 3\text{bits} + 8 \times 4\text{bits} = 193 \text{ bytes}$
- **Need to implement 256SQ uncoded bit slicer**

RS Overview

- **Reed-Solomon (RS) is a block based error correction system**
 - Based on symbols that are powers of prime numbers
 - Most popular is obviously $2^8 = \text{byte}$
 - These symbols can represent a Galois field which means that multiplication and addition work across this field
 - Allows the solution of equations that locate and identify errors
 - A Galois field based on 2^8 is called GF256, and can be used to make a Reed-Solomon block code containing up to 255 bytes

RS Overview (continued)

- **In a typical RS256 (i.e. RS code based on 2^8) implementation, there are a certain number of message bytes and check bytes**
 - Rule is that for every two check bytes ($2T$) you dedicate in the block, you can locate and identify one bad byte (T), and guarantee that you will detect up to $2T$ bad bytes (i.e. even though you can't correct them, you'll know they are bad)

Error Detection Probability

- In this proposal, we are talking about replacing the CRC-8 over the uncoded bits with the RS256(194, 192) code, which is roughly equivalent to a CRC-16 (2 check bytes = 16 bits), so the error performance is significantly better than CRC-8
- Error detection for the coded bits is maintained by the 325 LDPC parity bits, also much better than CRC-8.