Gigabit Ethernet
1000BASE-T

- Focus -

Physical Coding Sublayer (PCS)

Functional Basics and Overview

Based on: IEEE 802.3ab - Draft4.2 dated November 15 ,1998

March 29, 2000 Update: Corrected error in Trellis Diagram, added 1000T Convolutional Encoder and Convolutional Reset slides

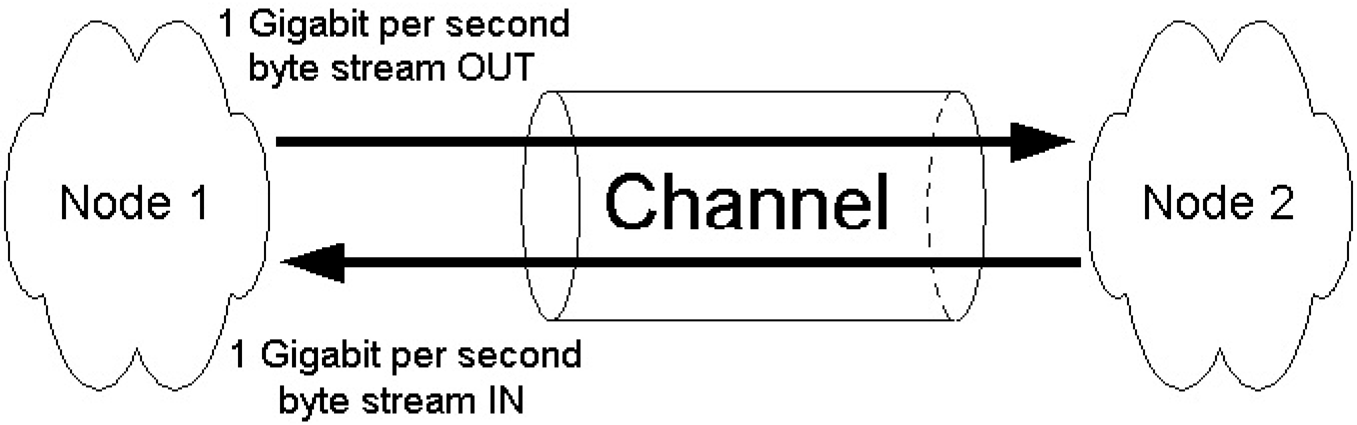
1000BASE-T Tutorial Series
IOL Gigabit Ethernet Consortium
prepared by Bob Noseworthy November 19th 1998

Presentation Outline

- Brief System Overview
- Basic Coding Theory Overview
- The 1000BASE-T Coding System

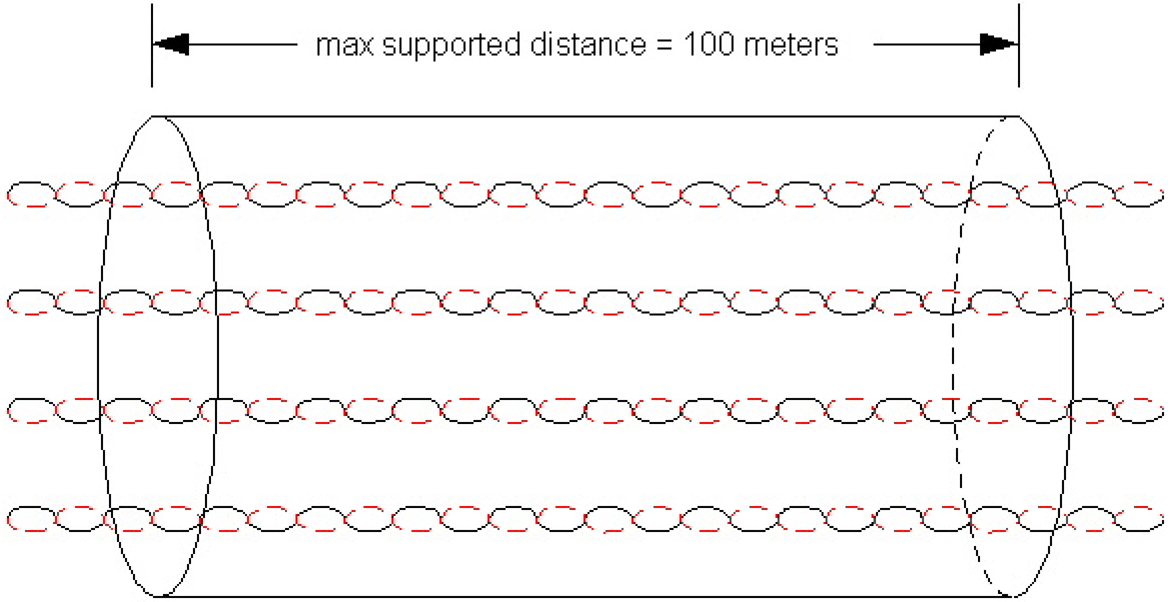
System Objectives

- Provide 1000Mbps Full Duplex Link between nodes



System Objectives

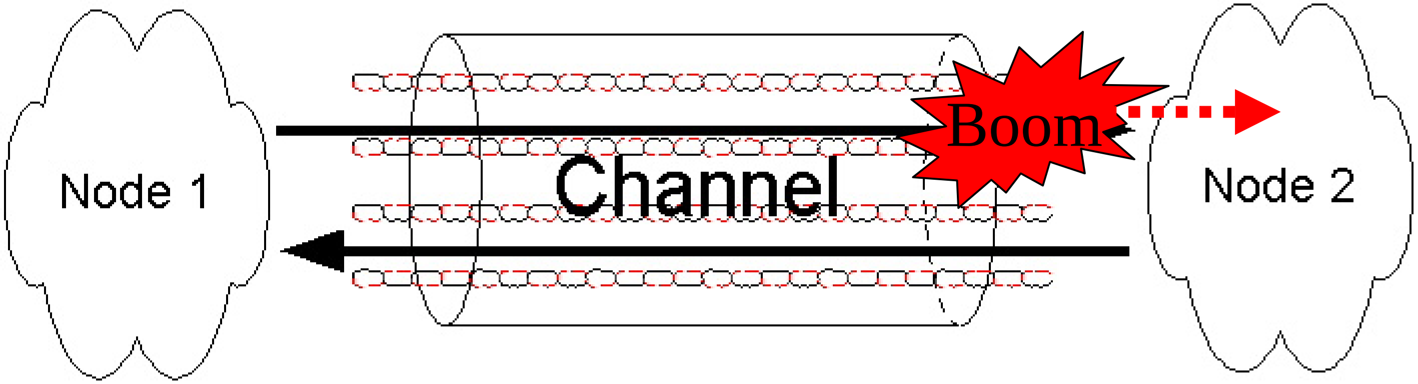
- Provide 1000Mbps Full Duplex Link between nodes
- Support large installed base of Cat 5 Balanced cable



Category 5 4-pair Balanced Cable

System Objectives

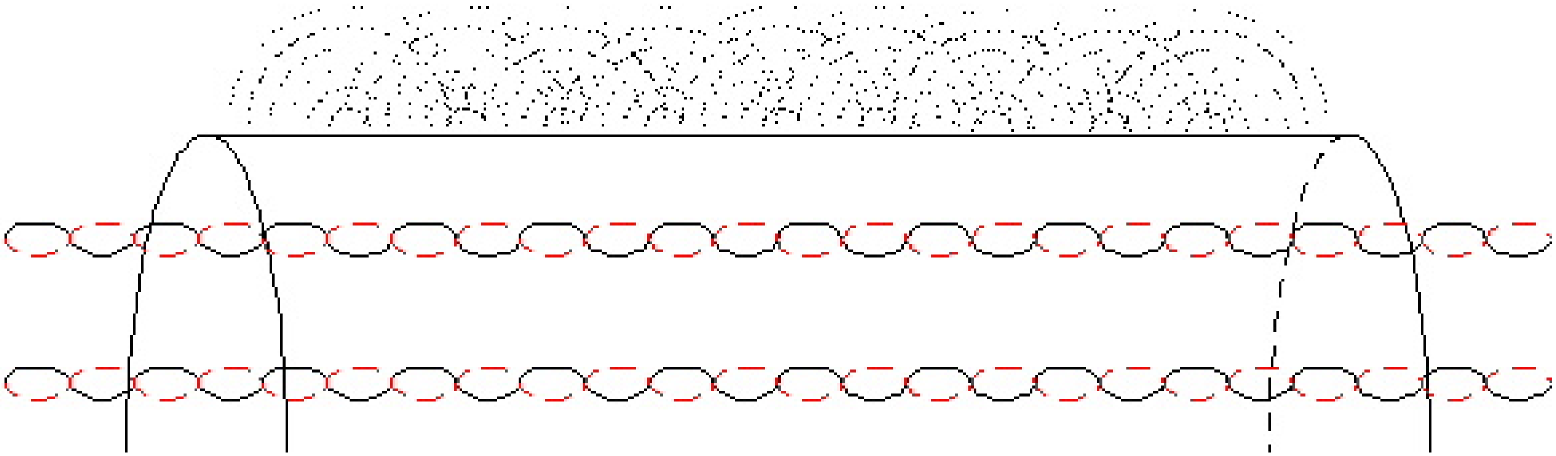
- Provide 1000Mbps Full Duplex Link between nodes
- Support large installed base of Cat 5 Balanced cable
- Operate with a bit error rate (BER) of $\leq 10^{-10}$



To put this in perspective: One byte is sent every 8ns. If bytes of data were sent constantly, every 10 seconds a bit detected at the receiver **may** be erred.

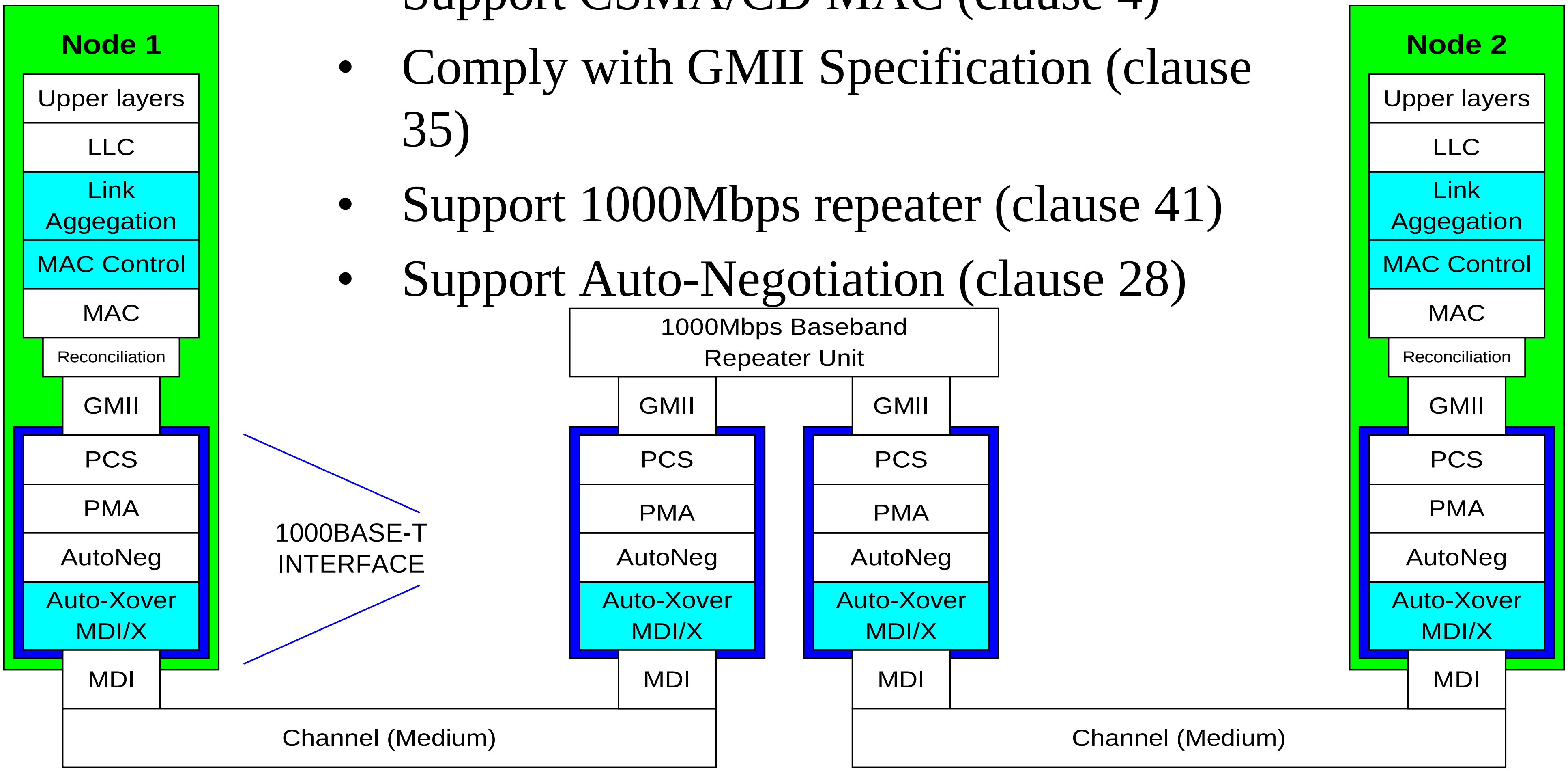
Objectives

- Provide 1000Mbps Full Duplex Link between nodes
- Support large installed base of Cat 5 Balanced cable
- Operate with a bit error rate (BER) of ≤ 10^{-10}
- Meet or exceed FCC Class A/CISPR



Other Objectives

- Support CSMA/CD MAC (clause 4)
- Comply with GMII Specification (clause 35)
- Support 1000Mbps repeater (clause 41)
- Support Auto-Negotiation (clause 28)



Basic Coding Theory Overview

- Communication Systems 101
- Scrambling Basics
- Descrambling Basics
- Block Codes
- Trellis Encoding
- Viterbi Decoding

Communication Systems 101

- What is the goal of any communication system?

Communication Systems 101

- What is the goal of any communication system?

  To convey information to a receiving party or parties.

  - Be it a radio transmission across a hostile battlefield environment
  - A laser beam between two distant satellites
  - Or simply a person trying to communicate to someone across a crowded noisy room

Communication Systems 101

- What is the goal of any communication system?
- What is required to communicate?

Communication Systems 101

Communication Systems 101

- What is the goal of any communication system?
- What is required to communicate?
  - Prior knowledge of rate constraints
  - Prior knowledge of the symbol set in use
  - Prior knowledge of the structure of what is being conveyed

Take for example: A conversation (say a presentation)

- A speaker can only talk so fast (or slow) and still be understood
- Phonemes make up the basic set of sounds used
- Those sounds form known words for a given:
  - **language**
  - **grammar**
  - **dialect**

Scrambling Basics

- First - What is the purpose of scrambling?

The purpose of scrambling is NOT to make a receiver's job more difficult (except maybe in military applications) but rather to combat issues related to the channel between the transmitter and receiver.

(recall that the term "channel" refers to any medium between the transmitter and receiver - be it copper cabling, fiber optic cabling, air or vacuum)

Scrambling Basics

- First - What is the purpose of scrambling?

The purpose of scrambling is NOT to make a receiver’s job more difficult (except maybe in military applications) but rather to combat issues related to the channel between the transmitter and receiver.

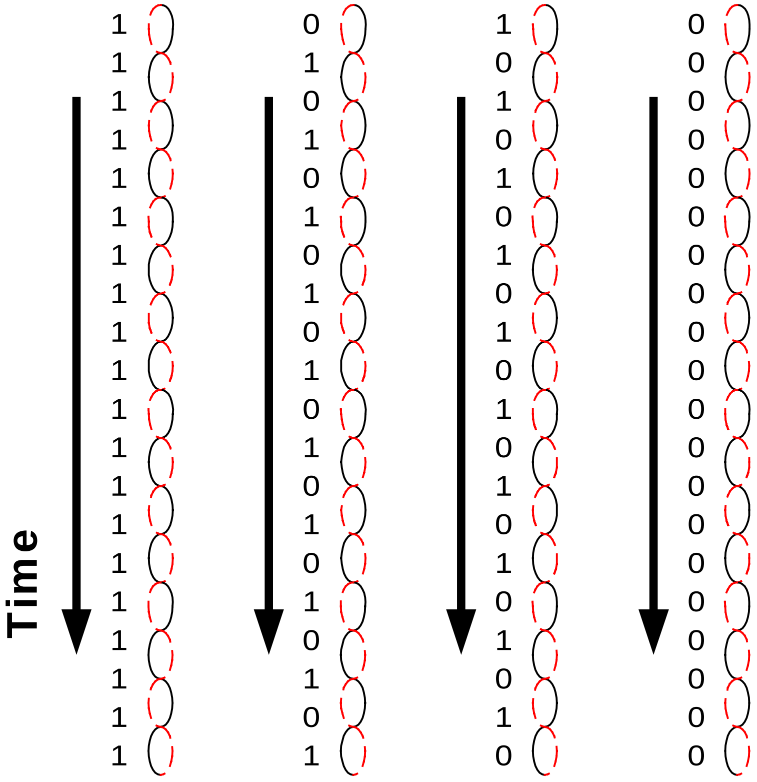
(recall that the term “channel” refers to any medium between the transmitter and receiver - be it copper cabling, fiber optic cabling, air or vacuum)

In 1000Base-T, scrambling’s main purpose is to temporally and spatially decorrelate the transmitted data. -- Lets explore that …

Scrambling Basics

- Why is scrambling necessary?

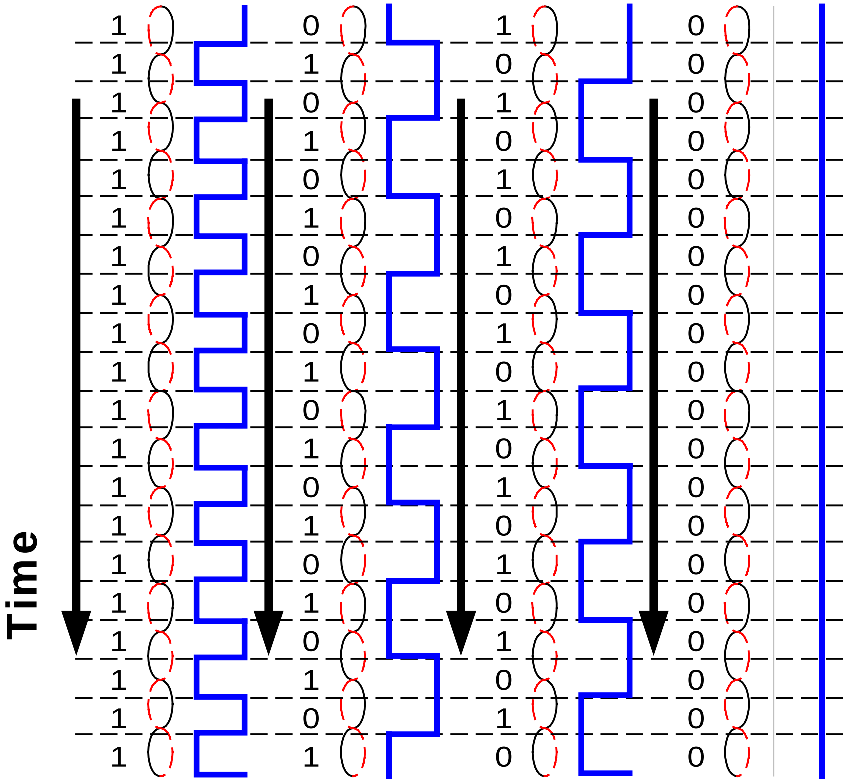
Consider a simple communication system that commonly sends a repeating pattern of 1010 and 1100 on a 4 channel medium.



1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0
1	0	1	0
1	1	0	0

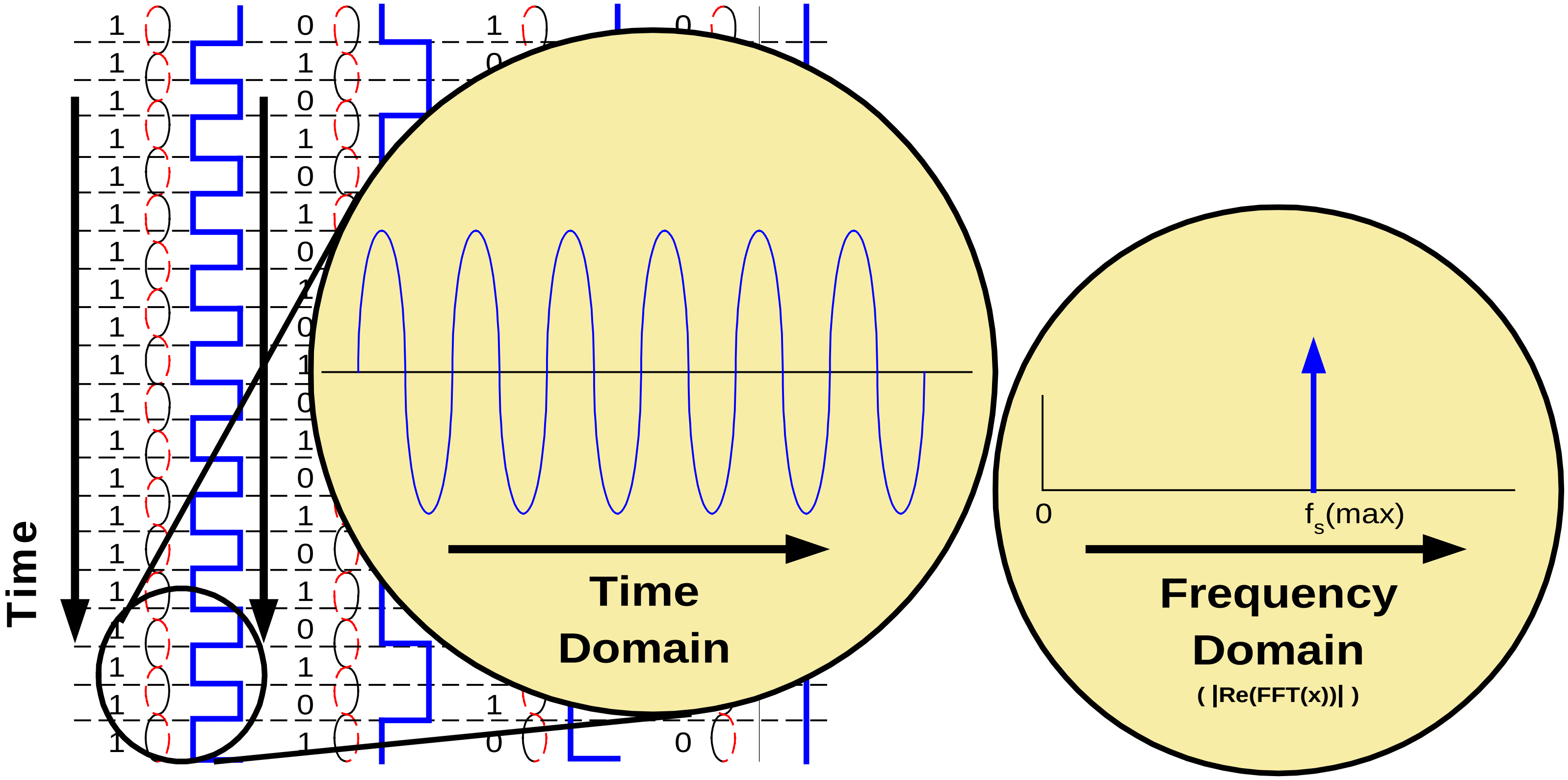
Scrambling Basics

Recall that “ones and zeros” are not sent on the channel, but rather an analog waveform is sent. In the diagram below a two level waveform is sent, where a data ‘1’ causes a transition, and a data ‘0’ causes the waveform to remain at the same level. (the horizontal dashed lines represent the bit-time boundaries)



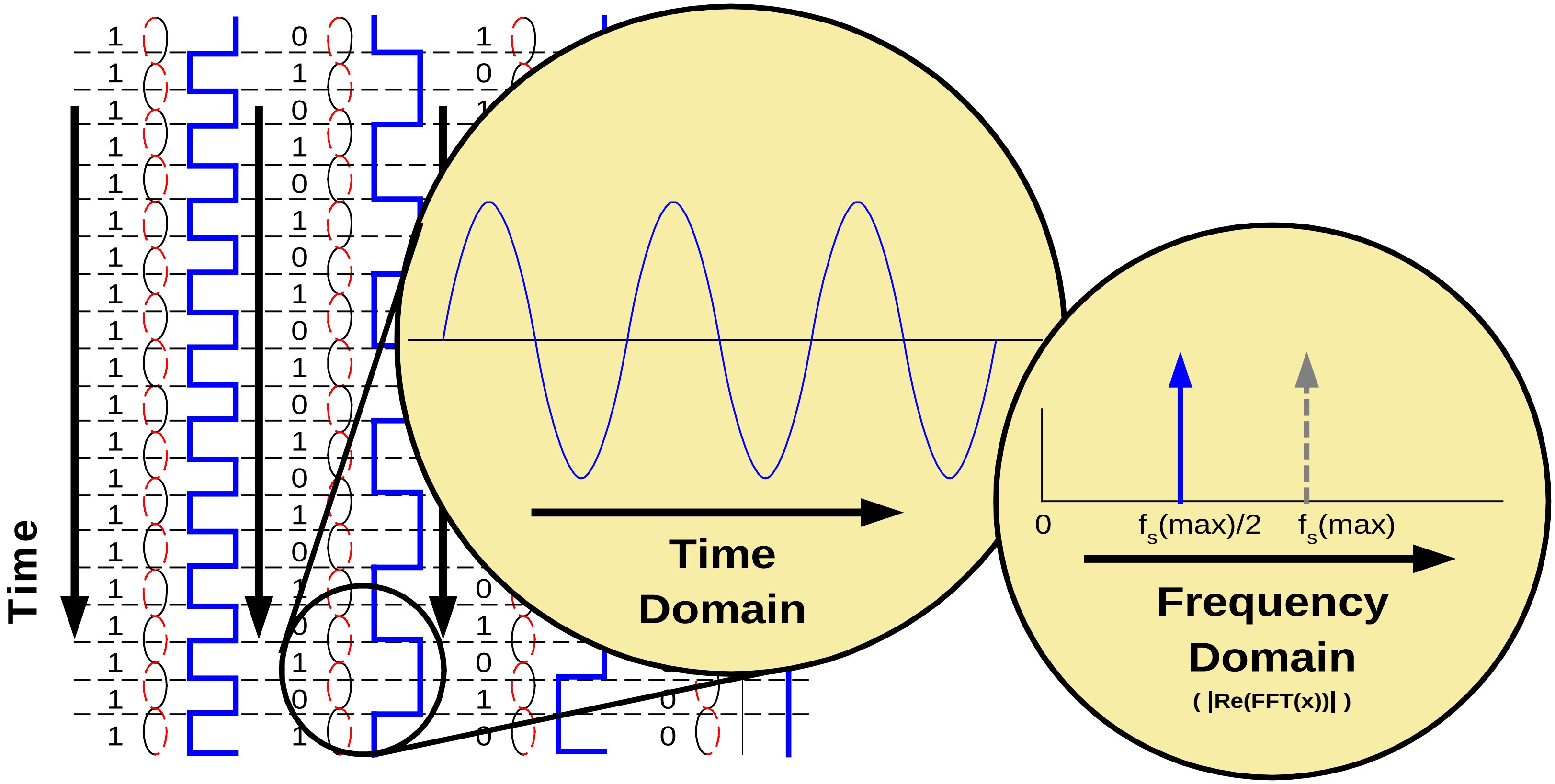
Scrambling Basics

Observe that the continuous repetition of 1's results in the highest frequency waveform on the channel. As higher frequencies tend to radiate "better", scrambling can help to eliminate such strong high frequency components (by temporally decorrelating the data on the channel.)



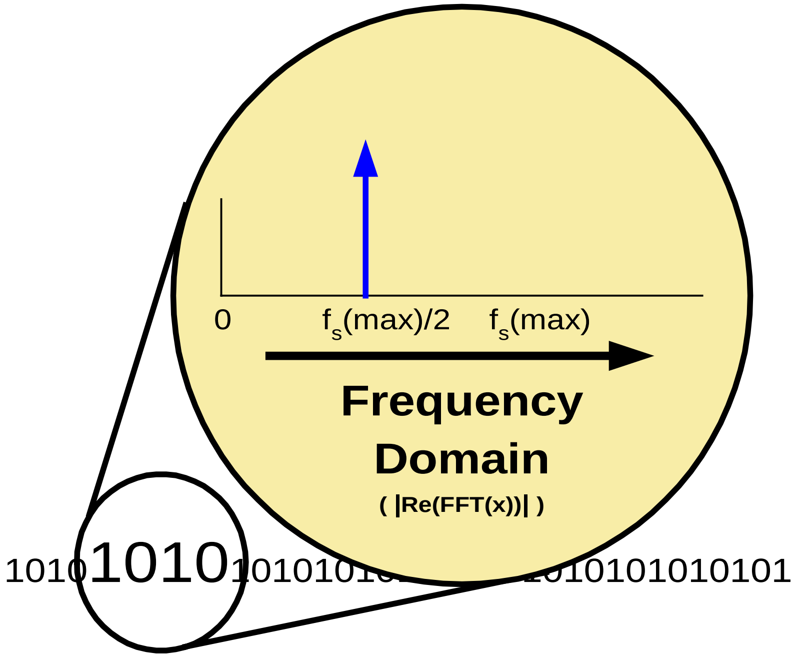
Scrambling Basics

In this example, as the “all 1’s” signal contains the highest frequency, any data sequence other than all 1’s naturally has a lower frequency content.

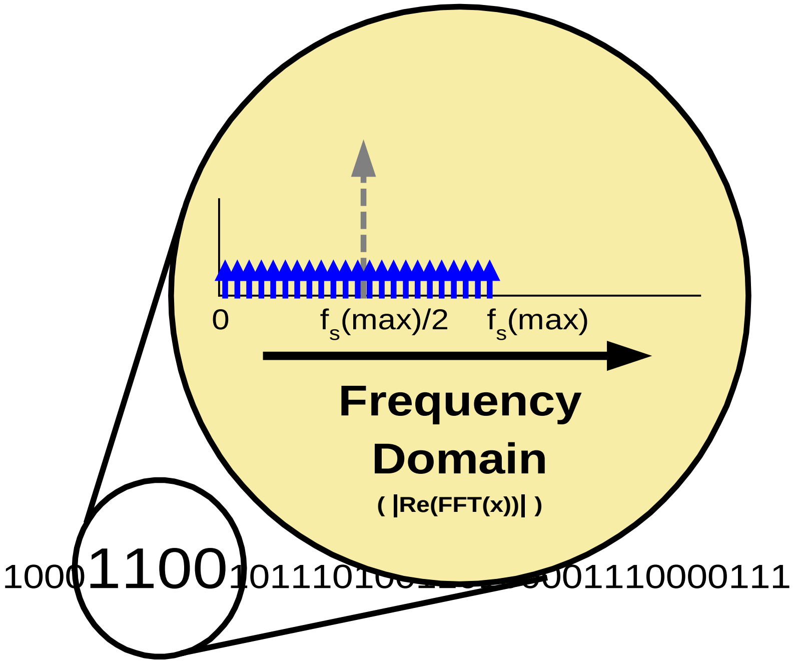


Scrambling Basics

By scrambling the data, no single frequency is sent for any significant period of time, thus the power is spread out over a range in the frequency spectrum. This type of technique is oft referred to as “spread spectrum” as it effectively whitens the frequency content of the signal and thereby reduces the power of any particular frequency component. This technique also makes more efficient use of the available bandwidth (as the entire band may be utilized).



Continuous Repeating 1010 pattern



Psuedo-random data pattern

Scrambling Basics

Scrambling is said to “whiten” the data’s frequency content based on the color white which contains all visible colors (frequencies).

Electrical noise is commonly Additive White Gaussian Noise (AWGN) which, simply put, means that the noise is random and occupies all frequencies (hence the term “white”).

By scrambling the data, the radiated power from the channel looks effectively like noise. This helps meet the FCC requirements. Also, since there are multiple channels in close proximity, the radiated power received on one channel from another (called cross-talk) is not correlated to the data being sent. This “spatial decorrelation” assists the receiver on the channel from distinguishing the desired signal from the background noise.

Scrambling Basics

- How do you scramble data?

Scrambling Basics

- How do you scramble data?

  Easy, take each symbol, and alter it by a random value!

Descrambling Basics

- How do you scramble data?

  Easy, take each symbol, and alter it by a random value!

- How do you scramble data such that it can be descrambled?

  Take each symbol and alter it by a pseudo-random value.

Descrambling Basics

- Take each symbol and alter it by a pseudo-random value.

Take for example the scenario where:

the user wishes to send: HELLO WORLD

the transmitter maps the characters to “symbols” suitable for “the channel”

H	E	L	L	O	W	O	R	L	D
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
08	05	12	12	15	23	15	18	12	04

Descrambling Basics

- Take each symbol and alter it by a pseudo-random value.

Take for example the scenario where:

the user wishes to send: HELLO WORLD

the transmitter maps the characters to “symbols” suitable for “the channel”

“scramble” the symbols by adding a pseudo-random value

H	E	L	L	O	W	O	R	L	D
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
08	05	12	12	15	23	15	18	12	04
-1	**-1**	**+1**	**-1**	**+1**	**+1**	**+1**	**-1**	**-1**	**+1**
07	**04**	**13**	**11**	**16**	**24**	**16**	**17**	**11**	**05**

Descrambling Basics

- Our example continues - at the receiver:

Assume the receiver detects the scrambled sequence properly:

If we decode this as is:

07 04 13 11 16 24 16 17 11 05

↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓

G D M K P X P Q K E

Clearly, this is not: HELLO WORLD

The receiver must somehow know what scrambling operation was performed on the data!!

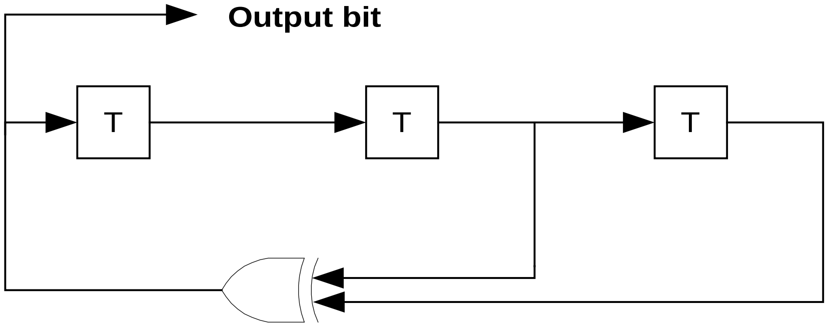
Descrambling Basics

- The receiver must know the scrambling sequence

  Hence why the scrambler must use a pseudo-random value - if it were truly random, the receiver could not know the sequence!

Descrambling Basics

- The receiver must know the scrambling sequence
  - Hence why the scrambler must use a pseudo-random value - if it were truly random, the receiver could not know the sequence!
- Pseudo-random values can be generated by any sequence which appears to be locally random, but is actually periodic.
  - A common practical implementation of a pseudo-random number generator is through a linear feedback shift register (LFSR).
  - An LFSR of ‘n’ elements will identically repeat its output every 2^n-1 outputs
  - A simple 3 element LFSR is shown below:



The repeating 7-code pattern for this LFSR is 0010111 (1 = “1” 0 = “-1”)

Descrambling Basics

- The receiver must also synchronize its descrambler

To recover the current location of the transmitter in its scrambling sequence.

One option to achieve this method, “brute force” exhaustive search of space - requires knowledge of what ‘should’ be sent (this is typically the idle pattern - say, HELLOWORLD was sent constantly between data)

```
07 04 13 11 16 24 16 17 11 05
-1 +1 -1 -1 -1 +1 +1 -1 +1 +1     F E L K O Y Q P L F
+1 -1 +1 -1 -1 -1 +1 +1 -1 +1     H C N K O W Q R K F
+1 +1 -1 +1 -1 -1 -1 +1 +1 -1 ->  H E L L O W O R L D
-1 +1 +1 -1 +1 -1 -1 -1 +1 +1     F E N K Q W O P L F
+1 -1 +1 +1 -1 +1 -1 -1 -1 +1     H C N L O Y O P K F
+1 +1 -1 +1 +1 -1 +1 -1 -1 -1     H E L L Q W Q P K D
-1 +1 +1 -1 +1 +1 -1 +1 -1 -1     F E N K Q Y O R K D
-1 -1 +1 +1 -1 +1 +1 -1 +1 -1     F C N L O Y Q P L D
-1 -1 -1 +1 +1 -1 +1 +1 -1 +1     F C L L Q W Q R K F
+1 -1 -1 -1 +1 +1 -1 +1 +1 -1     H C L K Q Y O R L D
```

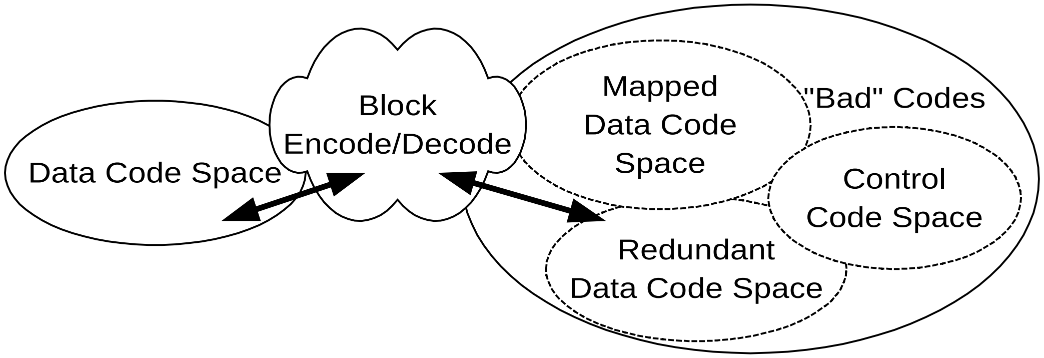
Descrambling Basics

Fortunately, real implementations transmit the current state of the LFSR (typically in the idle mode) such that the receiver can easily recover the state of the transmitter’s scrambler.

Rather than exhaustively searching for alignment, the receiver’s ‘n’-bit descrambling LFSR is simply “primed” with the first ‘n’ bits received from the line.

Block Codes

- Expands the code space of the data being transmitted.
- Allows an intelligent selection of channel symbols from the desired block of data being sent.
- Typical Benefits of Block Codes
    - Permits rich transition densities (allows for easier clock recovery)
    - Permits DC Balanced codes to be used
    - Permits non-data (control) codes, such as IDLE, Start of Frame, etc
- Some Ethernet Block Codes: 4B/5B 8B/10B 6B/3T



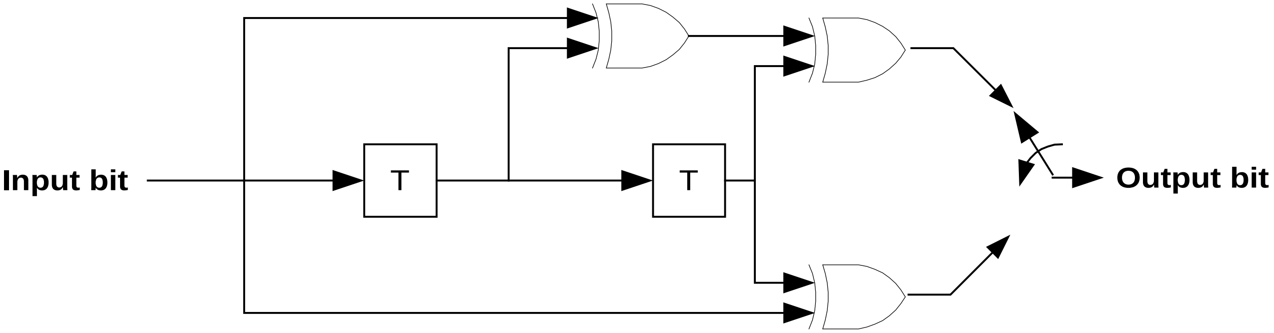
ex: 100Base-TX 4B/5B “0 0 0 0”   <-----> “1 1 1 1 0”

Convolutional Codes

- Convolutional Codes could be considered a special class of Block Codes
- The term “convolutional” is used as the output symbol sequence is generated by the convolution of the input sequence and a “generator” sequence.
- The generator sequence is a K element delay line with modulo-2 adder feedback. -- This is just another way of saying a LFSR!
- In terms we are familiar with, a convolutional code is simply the result of XOR’ing the transmit data with the output of a scrambler.

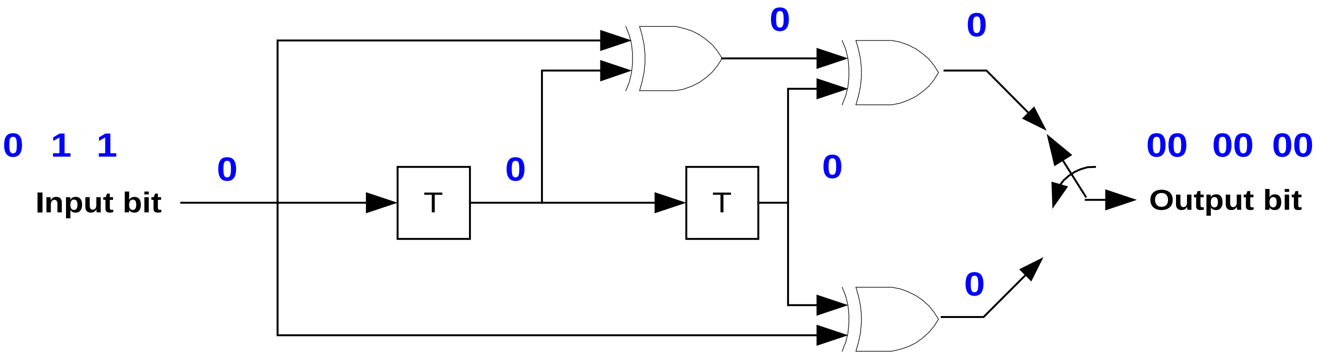
Convolutional Codes

- A Simple Convolutional Encoder example

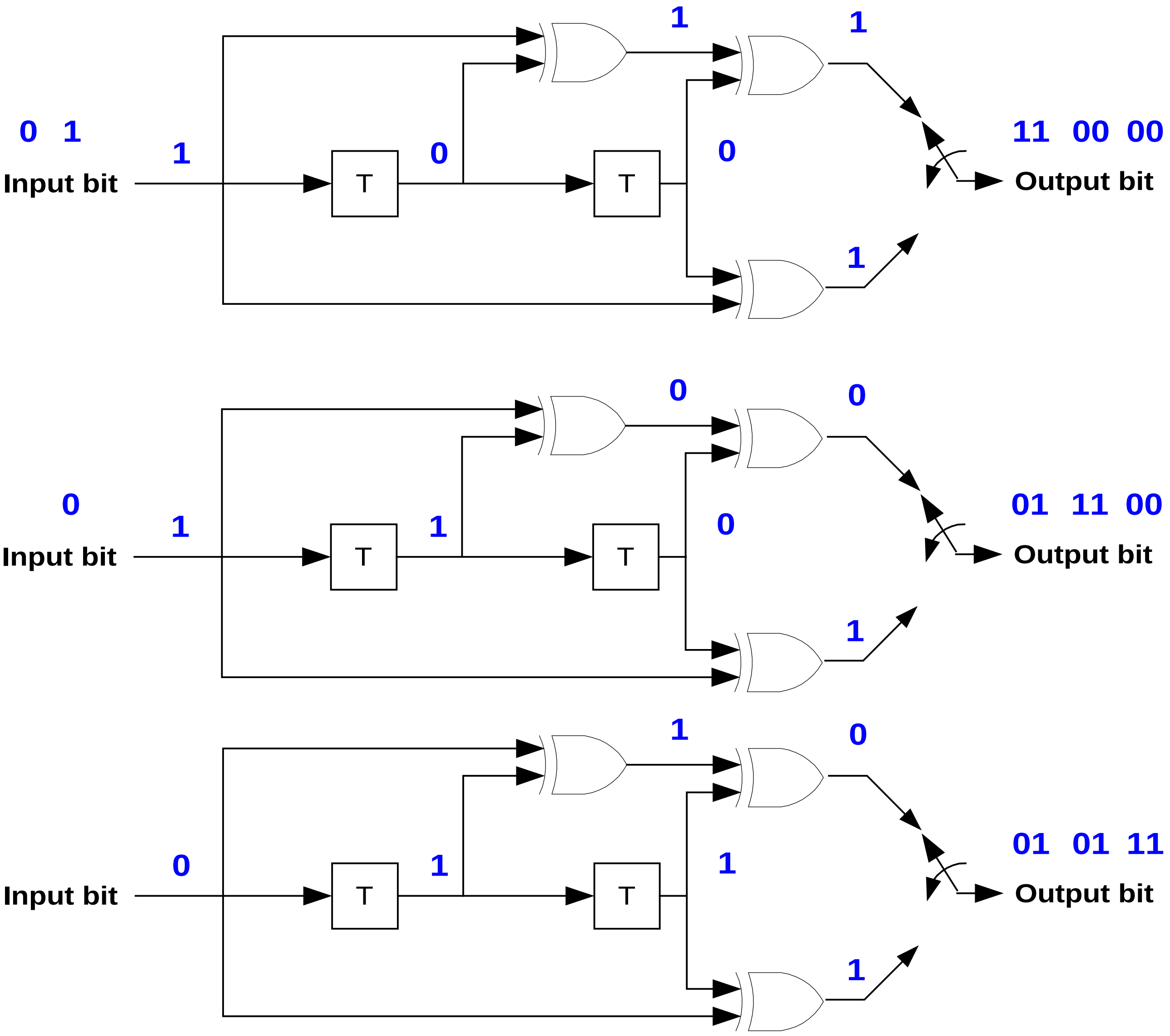


Below, the sequence $0_n1_{n-1}1_{n-2}0_{n-3}$ is fed into the encoder, consisting of the time delay blocks, and XOR blocks.

Note the output $00_n11_{n-1}01_{n-2}01_{n-3}$ is at twice the input rate



Convolutional Codes

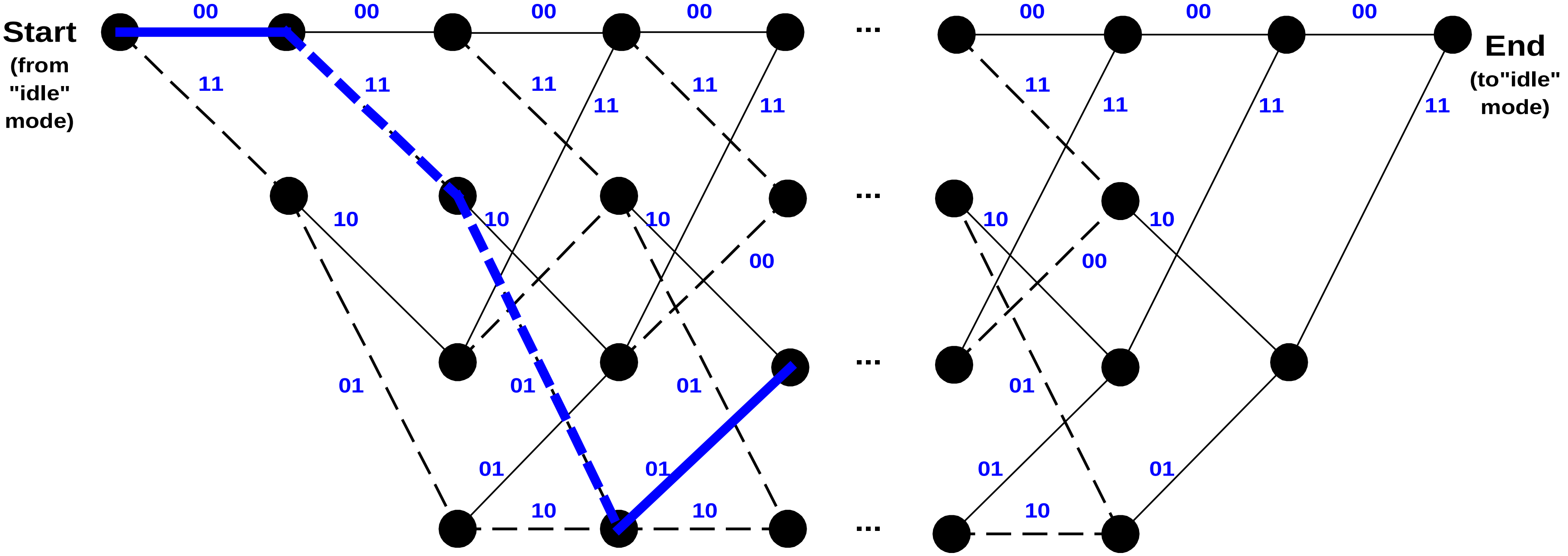


Trellis Diagrams

- The preceding convolutional encoder can be represented in another form - a Trellis Diagram.

Trellis Diagrams

- Recall, the Convolutional Encoder example encoded 0110 to 00,11,01,01.



If data to send is '0', follow solid line from state and output codegroup (in blue)
If data to send is '1', follow dashed line from state and output codegroup (in blue)

Trellis Diagrams

- It should be clear now how this simple trellis provides a “structure” to the transmitted data stream, as only valid transitions through the trellis may be transmitted!

(In the example, after the codegroup 11 is sent, only the codegroup 01 or 10 are permissible)

Viterbi Decoding

- This structure provided to the underlying symbols transmitted is comparable to spelling and grammar rules. consider:

  “I coldn’t wait til it was over”

  “I can’t believe their still awake”

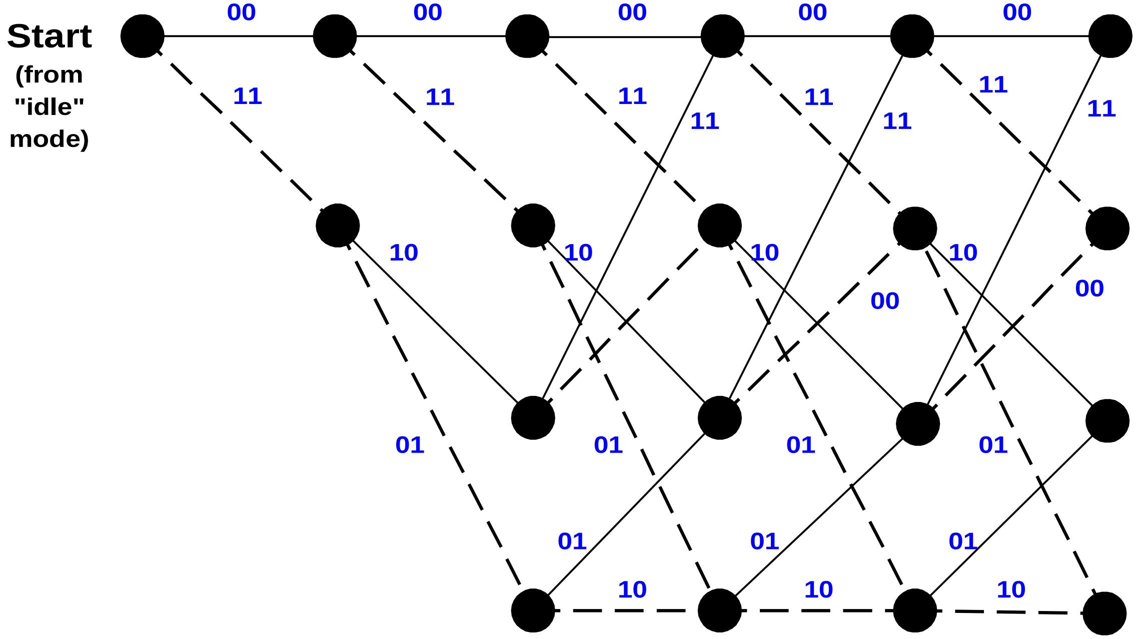
  Both symbol sequences are erred, but the knowledge of the structure of the transmission, allows the receiver to properly decode the sequence.

Viterbi Decoding

- A Viterbi Decoder provides error correction. Not just error detection like most other LAN block codes.
- This results in a tolerance to a certain bit error rate (BER). Thus, the overall system performance, often expressed in terms of the signal-to-noise ratio (SNR) is effectively increased by several dB when a trellis encoder/viterbi decoder is employed.

Viterbi Decoding

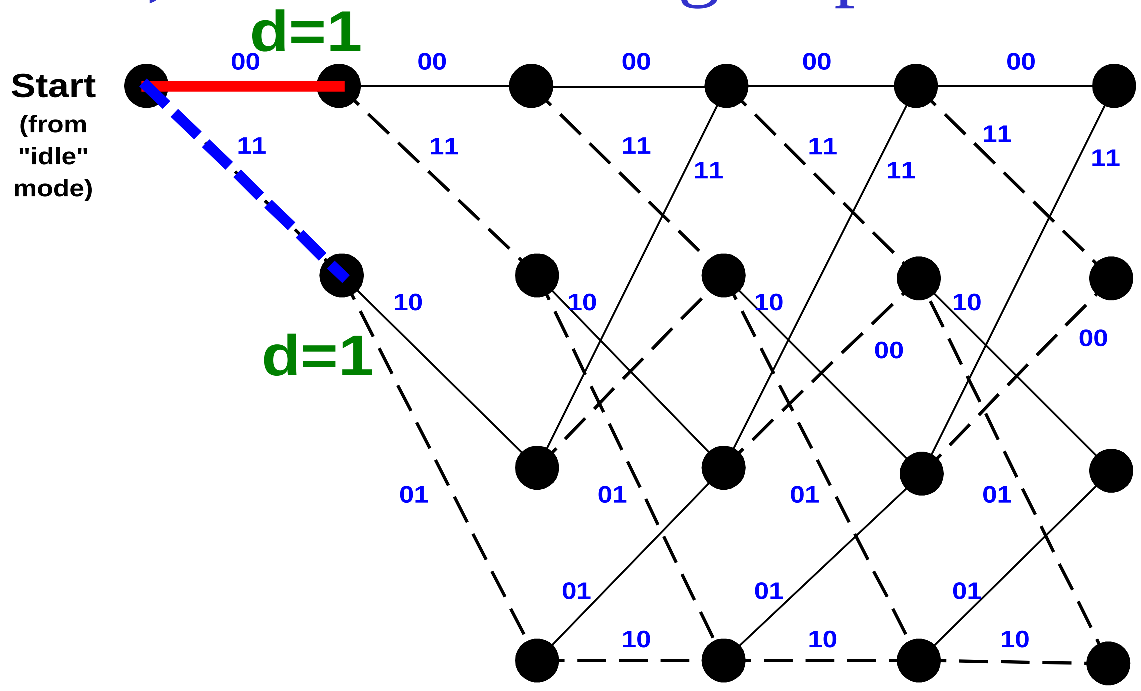
- Consider once again the example trellis diagram



If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- From idle, the first code group is received

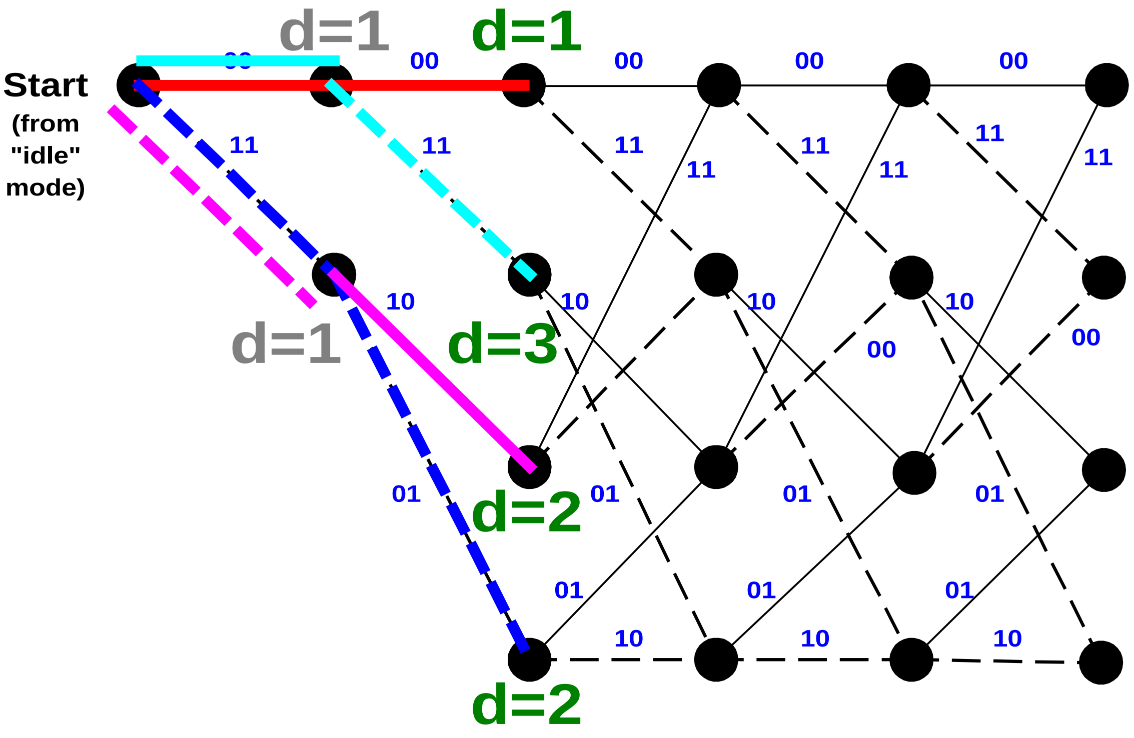


Received Data **01**

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- As more codes arrive, multiple possible paths emerge

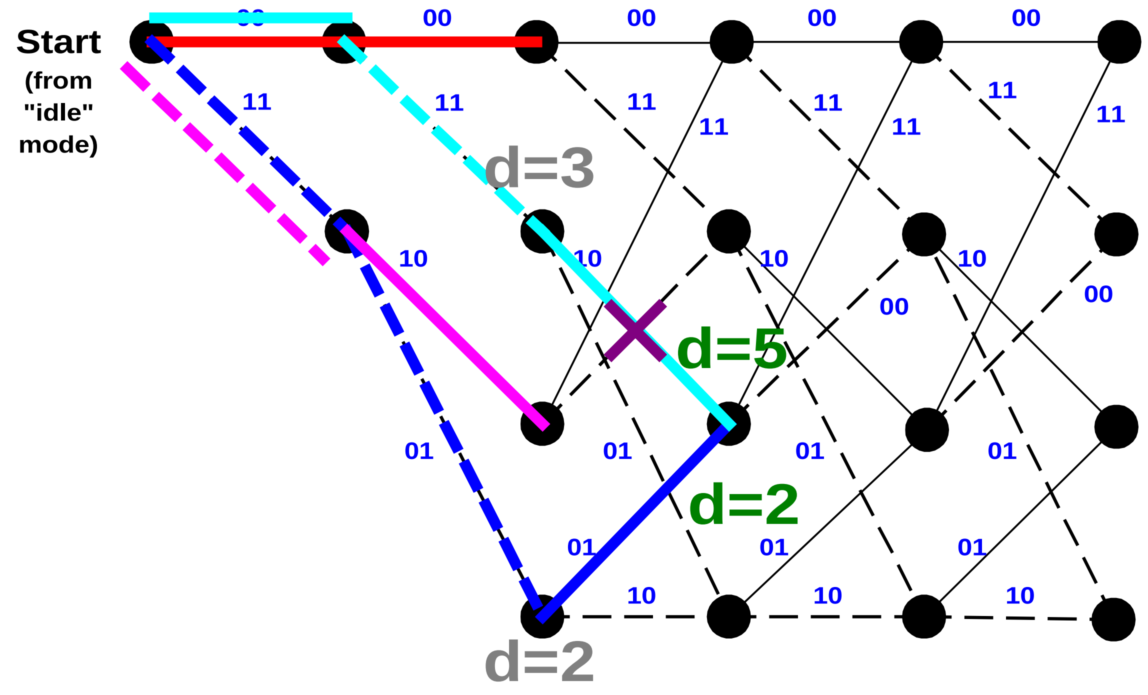


Received Data **01** **00**

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- Some of these paths merge, leaving only 1 survivor

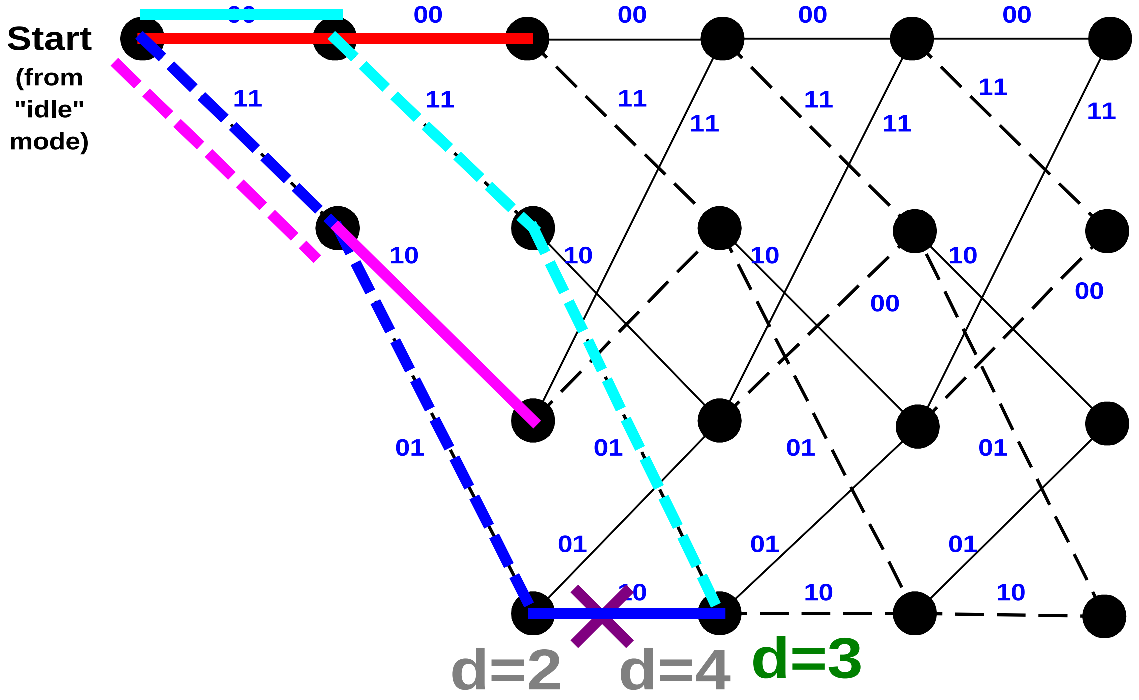


Received Data 01 00 01

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- As merged paths represent higher/(or equal) cost paths to the same point, no harm is done by eliminating them.

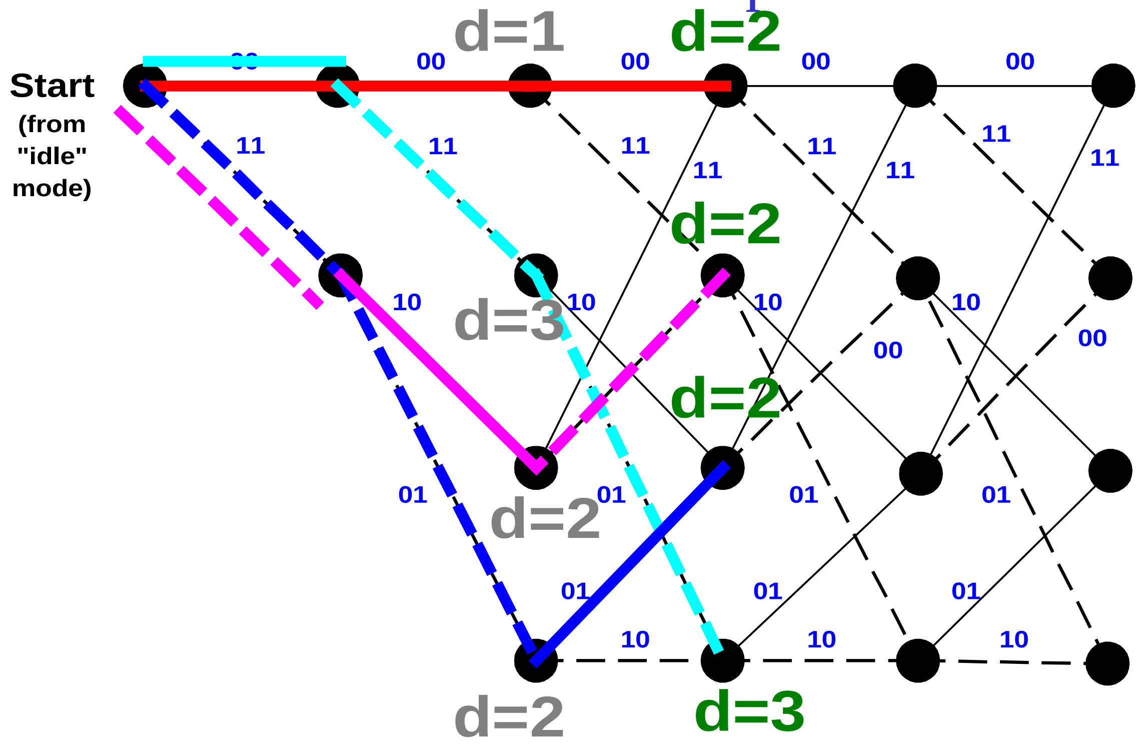


Received Data **01** **00** **01**

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- As more codes arrive, the overall path costs change. How many codes must the system wait for to achieve an optimal decode???

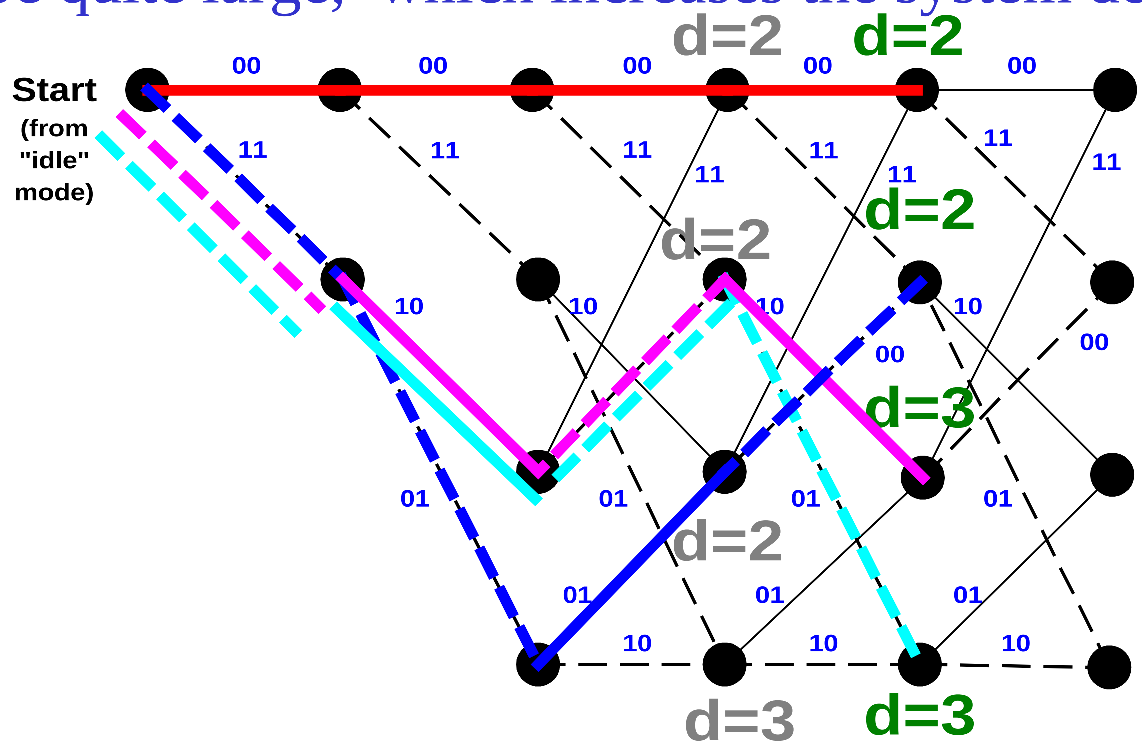


Received Data **01** **00** **01**

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- Depending on the trellis design, the required number of codes to wait for could be quite large, which increases the system delay!

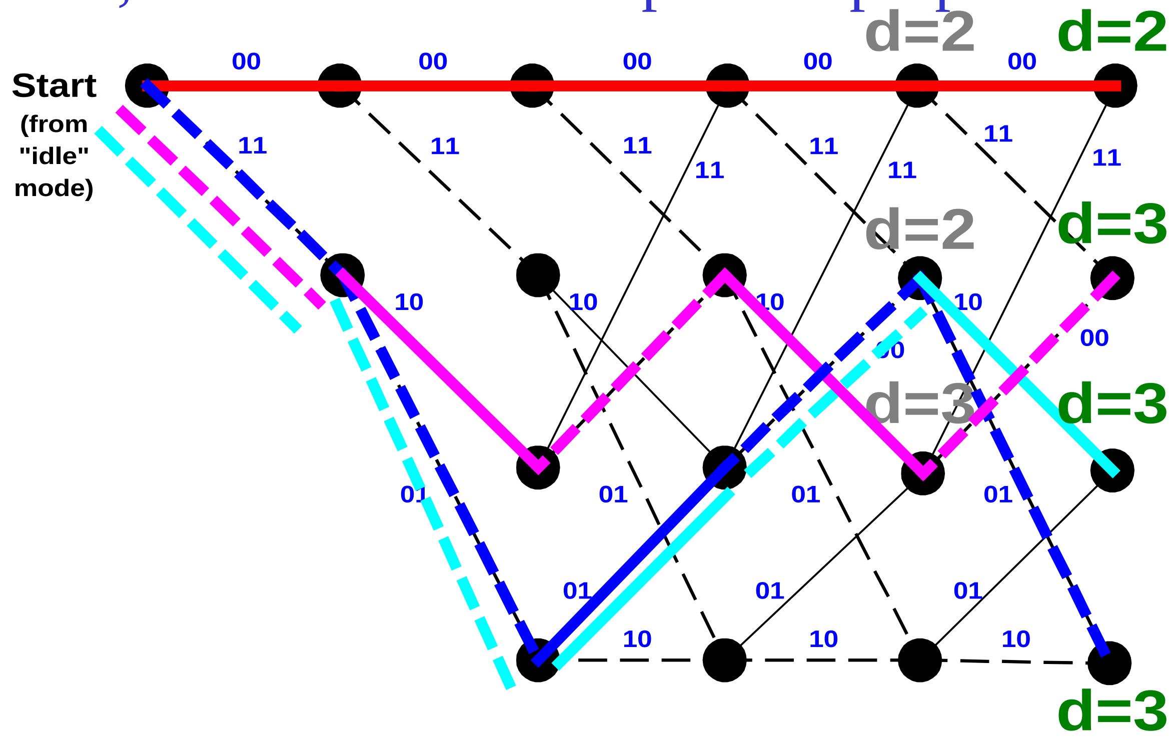


Received Data 01 00 01 00

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Viterbi Decoding

- Eventually, 0100010000 is decoded via the red path yielding the data 0,0,0,0,0. Thus, 2 bit errors did not prevent proper decode!!



Received Data 01 00 01 00 00

If path selected is a solid line, the received data is '0'
If path selected is a dashed line, the received data is '1'
The distance metric employed is a simple binary hamming distance
ie: '00' is 2 distant from '11', as is '01' from '10', etc..

Intermission

- Catch your breath
- Ask some questions that can’t wait
- Fasten your seatbelt, the fun is just beginning!!

The 1000BASE-T Coding System

- Selecting the Channel Symbols
- Increasing Symbol Distance
- Defining the 4D/PAM5 Structure
- The Trellis structure used
- The Master/Slave Scramblers employed
- The 1000BASE-T Frame Structure

Selecting the Channel Symbols

- The 1000BASE-T phy will interface to the upper layers via the 8-bit wide GMII. Thus, during frame transmission, the phy receives a new 8-bit word to send from the GMII every 8ns (125Mhz).
- A natural choice is to encode the full 8-bit word to some symbol space.
- To keep the signaling rate low, all four pairs of the Cat 5 cable will be used. Thus, only four 250Mbps channels are required.

Selecting the Channel Symbols

- A system could be selected that has a high number of levels per symbol period, however additional levels increase the complexity of the system.
- As the data from the GMII is clocked at 125MHz, a natural choice is to use this clock rate to drive symbols onto the channel - especially as 100Base-TX devices currently operate at 125MHz while sourcing three-level (MLT-3) symbols.
- Unfortunately, the phy is receiving 256 (2^8) data codes every 8ns. Even if the 3 level 100Base-TX system were used across all 4 pairs, only 81 (3^4) symbols would exist

Selecting the Channel Symbols

- So, while there are benefits to leveraging the 100Base-TX design knowledge at the 125MHz speeds, clearly a three level system will not work.
- Will 4 levels do? $4^4 = 256$, so yes, there are enough symbols to map the data codes; however, there are no remaining codes for control signals (idle, start of frame, end of frame) nor are there and codes available for redundancy.
- Will 5 levels do? $5^4 = 625$. Yes! In fact, enough codes exist in the symbol space to allow for 100% redundancy (use 512 codes for data, rather than just 256) and still leave 113 symbols for control signals.

Selecting the Channel Symbols

- These 5 symbols are labeled as -2, -1, 0, +1, +2

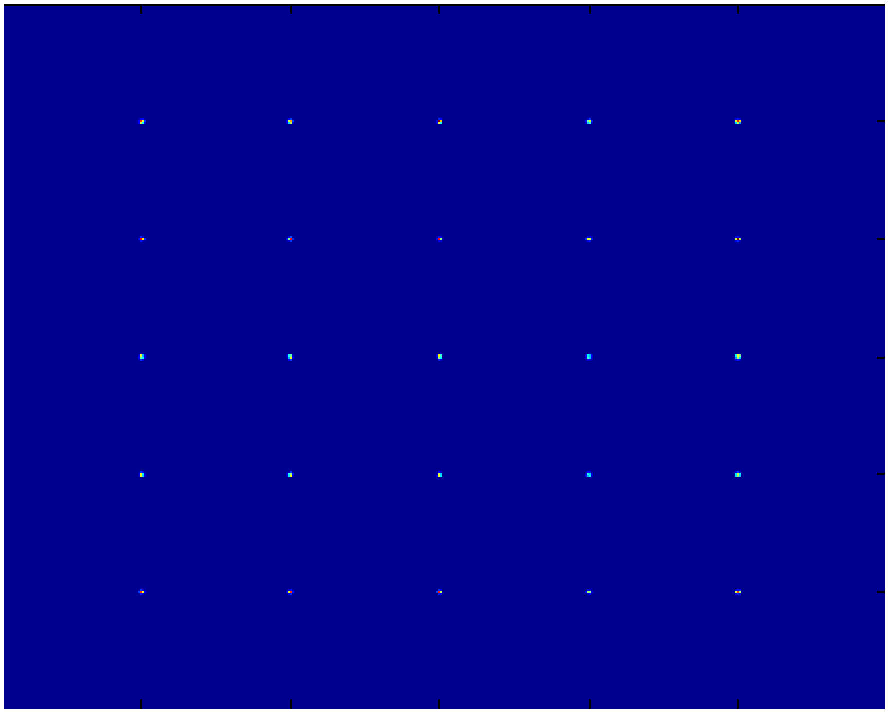
  (+/- 2 actually maps to +/-1V, and +/-1 maps to +/- 0.5V)

- If the symbols -1 and +1 are not used, and only one channel is transmitted on, then the output is very similar to 100Base-TX signaling (allowing for simpler dual 100/1000 speed implementations)

Increasing Symbol Distance

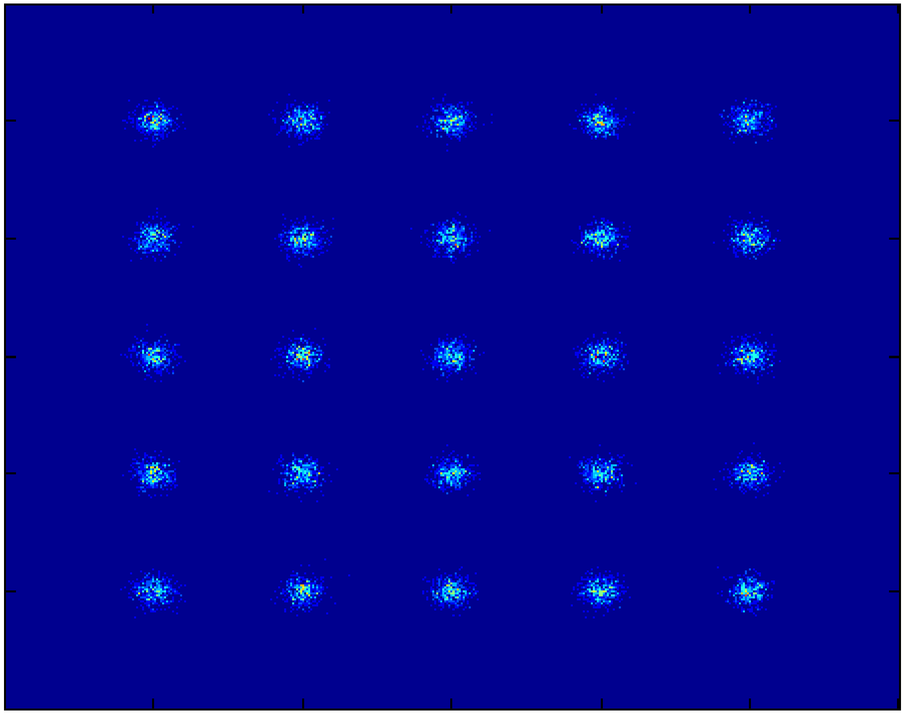
- While the symbols -2, -1, 0, +1, or +2 are sent per transmitter, recall that the 1000BASE-T system uses all four pairs of the Category 5 balanced cable.
- The combined output of all four transmitters on each line forms a four-dimensional constellation.

Such a constellation is formed by taking the possible outputs of each transmitter as an axis orthogonal (at right angles) to the other axis. For every symbol period, one point in the constellation is sent. While the real system is a 5x5x5x5 constellation, a simple 5x5 constellation is easier to explain…



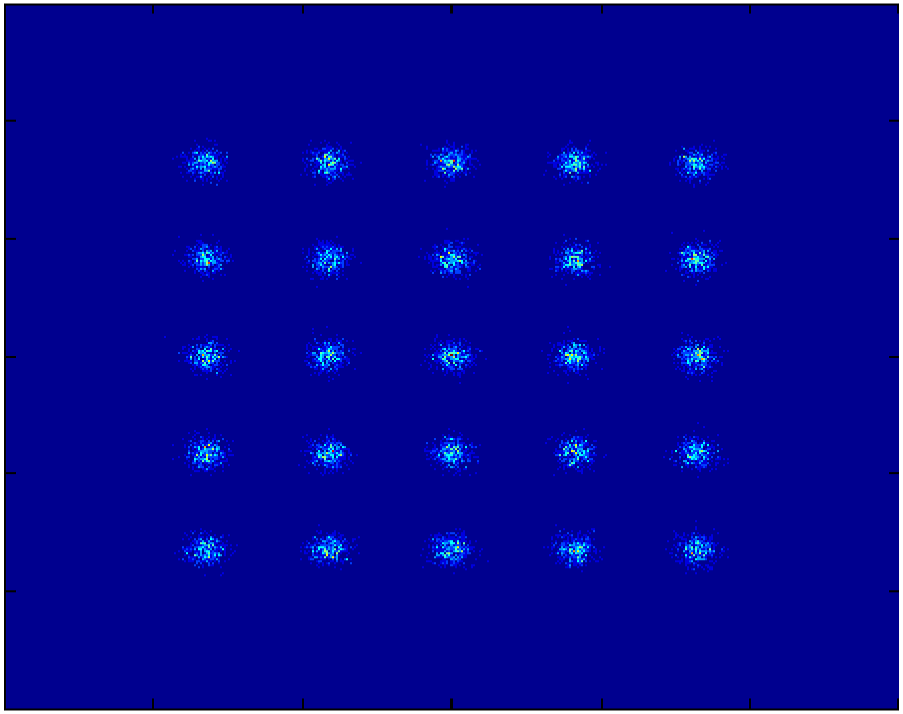
Increasing Symbol Distance

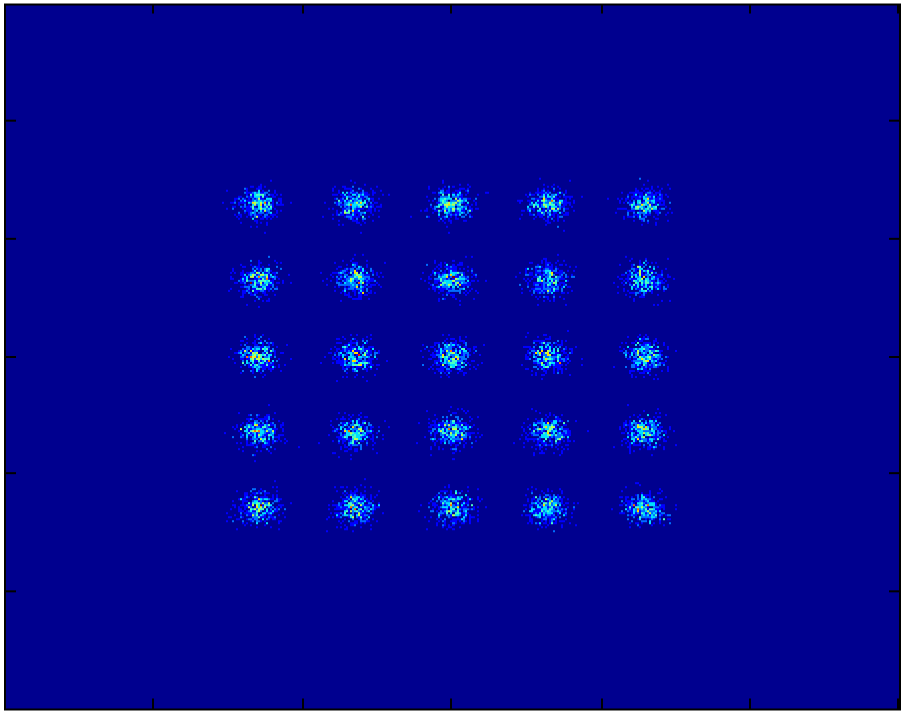
- However, noise affects the constellation. If too much noise disrupts the signaling, then the receiver will not be able to distinguish between the correct point in the constellation and neighboring (incorrect) points.



Increasing Symbol Distance

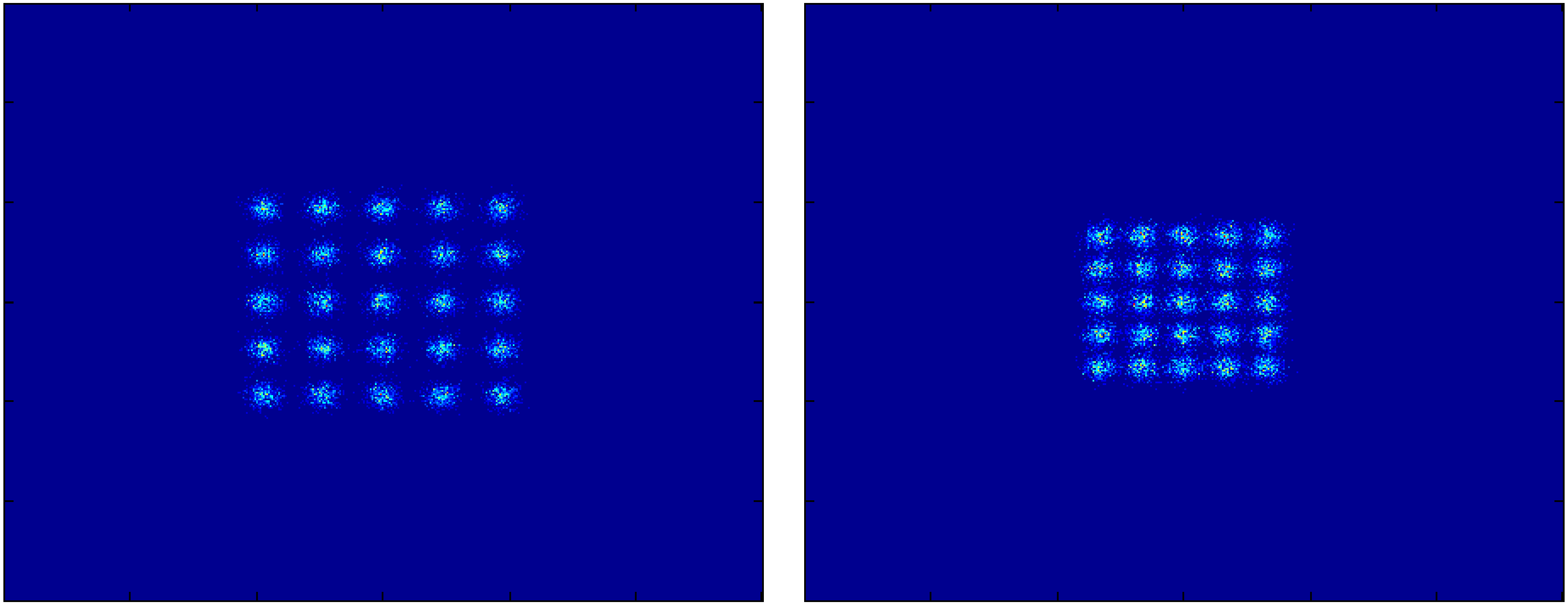
- Attenuation also affects the constellation, with similar concerns.





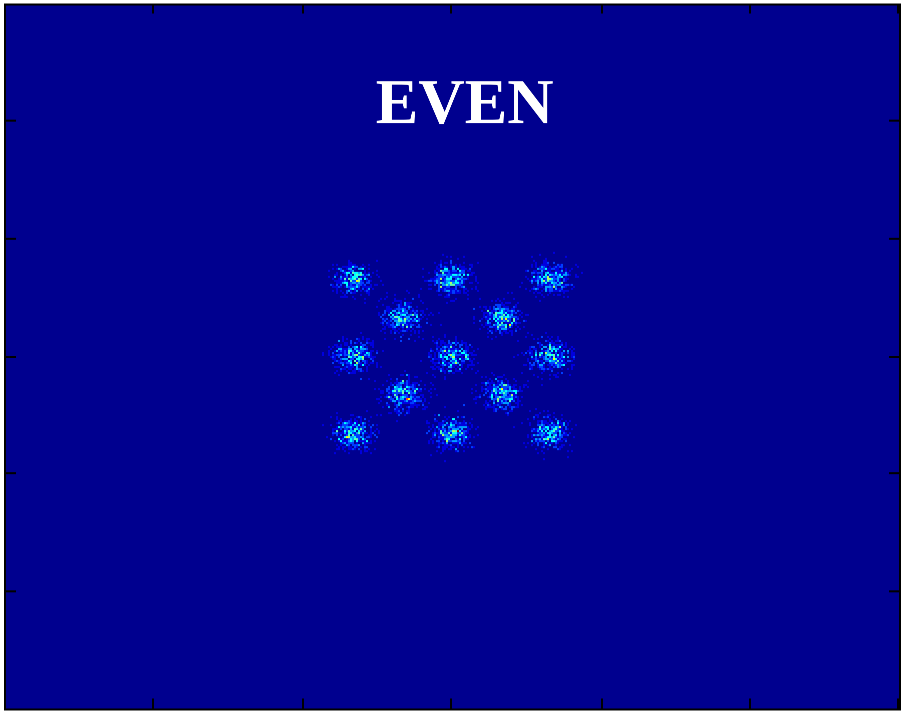
Increasing Symbol Distance

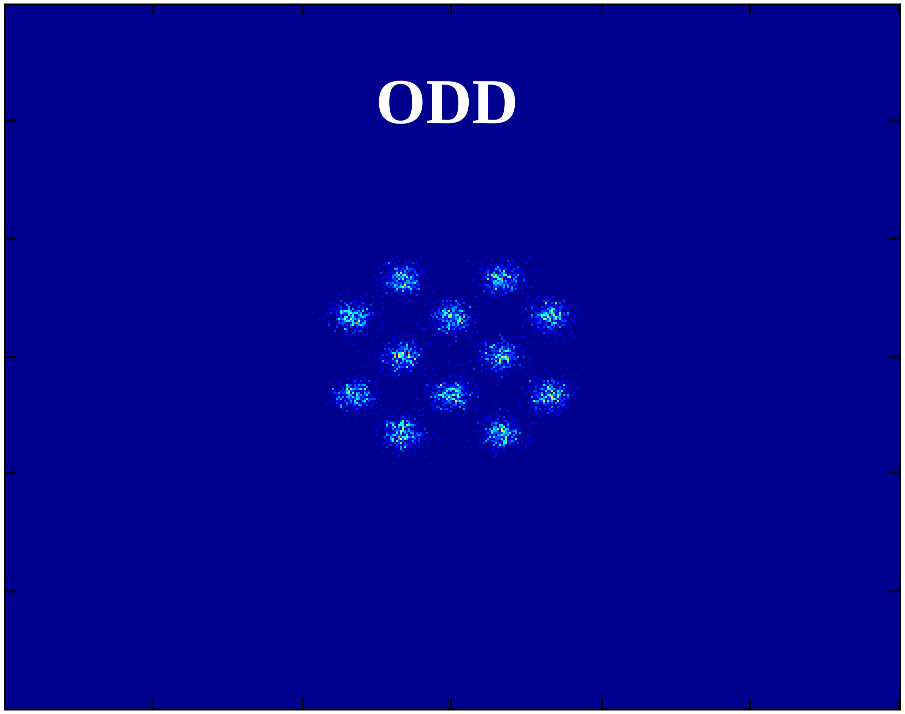
- As the attenuation/noise distortion increases, the problem of differentiating between points within the constellation becomes obvious.



Increasing Symbol Distance

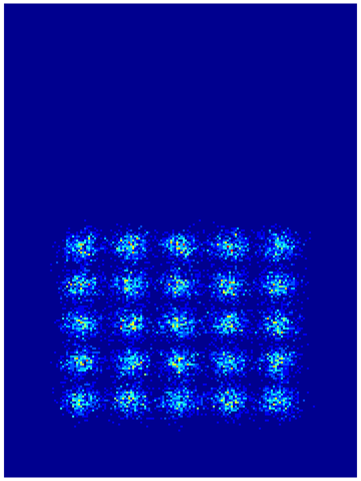
- To combat this problem, limit the permissible combination of symbols.
- The images below are identical to the previous, simply the “even” and “odd” constellation points have been separated.

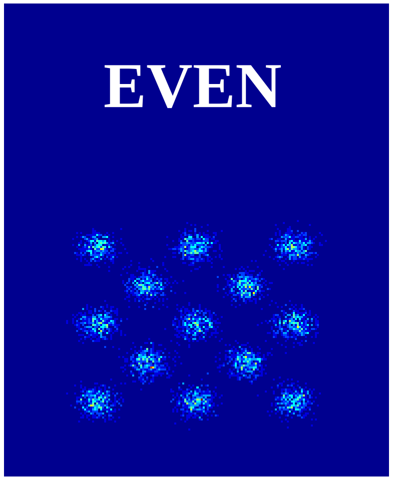


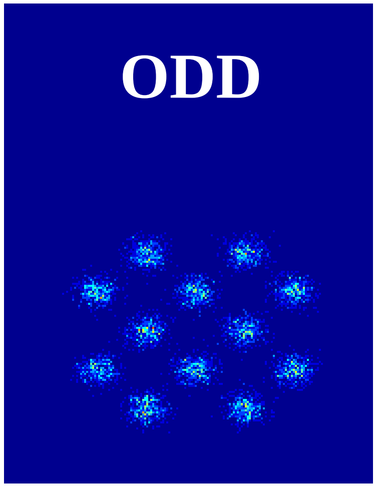


Increasing Symbol Distance

- The separation of “even” and “odd” points has the effect of increasing the minimum distance between the permissible transmit codes. Prior to the separation, the minimum Euclidean distance was 1. Now it is $2^{.5}$ Thus, the minimum squared Euclidean distance is now 2.







The 4D/PAM5 System

- Let us define symbol subsets X = -1, +1 & Y = -2, 0, +2
- The Even subset can be written as:

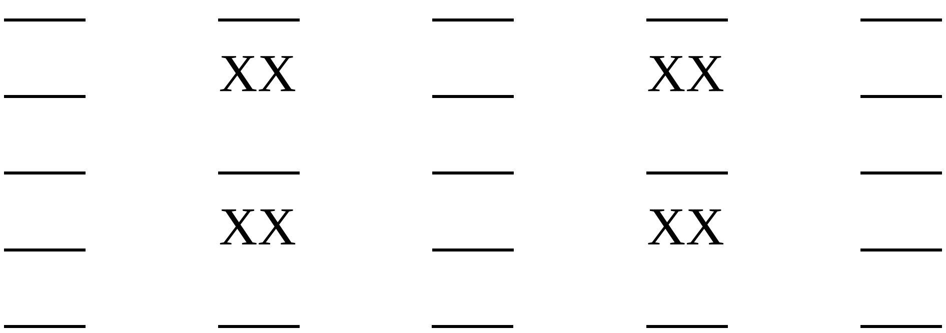
YY	___	YY	___	YY	
___	XX	___	XX	___	
YY	___	YY	___	YY	
___	XX	___	XX	___	
YY	___	YY	___	YY	min sqr distance = 2

This subset can be further separated to:

YY	___	YY	___	YY	
___	___	___	___	___	
YY	___	YY	___	YY	
___	___	___	___	___	
YY	___	YY	___	YY	min sqr distance = 4

The 4D/PAM5 System

The remaining subset from the even subset is:



min sqr distance = 4

The 4D/PAM5 System

- Likewise, the Odd subset can be written as:

___	XY	___	XY	___	
YX	___	YX	___	YX	
___	XY	___	XY	___	
YX	___	YX	___	YX	
___	XY	___	XY	___	min sqr distance = 2

This subset can be further separated to:

___	___	___	___	___	
YX	___	YX	___	YX	
___	___	___	___	___	
YX	___	YX	___	YX	
___	___	___	___	___	min sqr distance = 4

The 4D/PAM5 System

- Likewise, the Odd subset can be written as:

___	XY	___	XY	___	
___	___	___	___	___	
___	XY	___	XY	___	
___	___	___	___	___	
___	XY	___	XY	___	min sqr distance = 4

- Why is any of this useful??

Recall that the Viterbi Decoder will choose the lowest cost path, and even properly choose the correct path even after several error events have been received, provided no other (incorrect) path is a lower cost! Hence, maximizing the separation of the symbols is highly desirable.

The 4D/PAM5 System

- Recall that the symbol distance example up to this point has been for only a 2-D constellation. As each of the four channels can be sending a symbol from X or Y, there are naturally 16 combinations.
- These 16 sets can be reduced to 8 and still keep a min. sqr distance of 4 between the symbols on the four channels. This is done identically to the previous 2-D channel constellation sub-set breakdown. We can now form the 4D sub-lattices constellations:

D0	XXXX + YYYY
D1	XXXY + YYYX
D2	XXYY + YYXX
D3	XXYX + YYXY
D4	XYYX + YXXY
D5	XYYY + YXXX
D6	XYXY + YXYX
D7	XYXX + YXYY

Any two points within each sub-lattice has a min. square distance of 4.

Similarly, any two points in different sub-lattices has a min. square distance of 4.

The Trellis Structure

- ICBST (It can be shown that) the distance between any two points in a sublattice is at least 4.

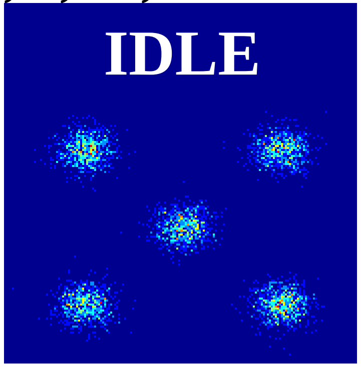
D4: XYYX + YXXY - pick +1,-2,+2,-1 and 0,-1,+1,0

$(+1 - 0)^2 + (-2 - -1)^2 + (+2 - +1)^2 + (-1 - 0)^2 = 1+1+1+1=4$

Of course, some points are more than 4 distant

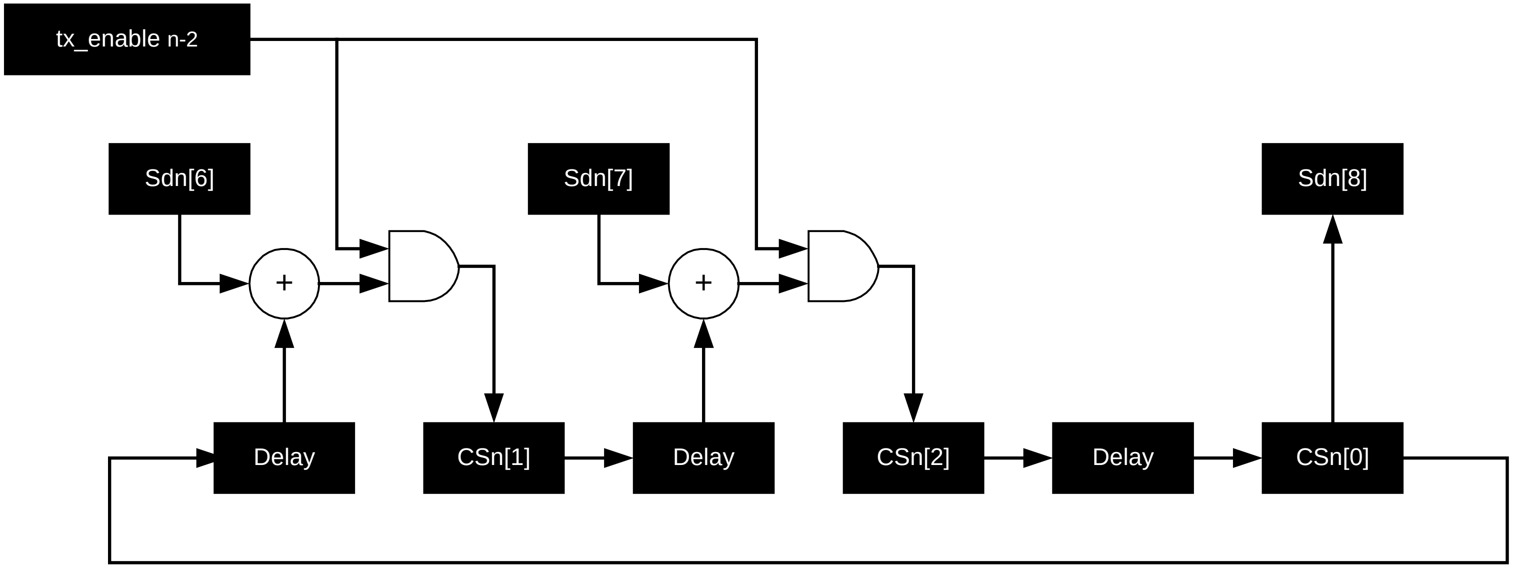
D0: XXXX + YYYY pick -2, -2, 0, -2 and 0,0,-2,0 dist=8 !

D0 (YYYY specifically) make up the 1000BASE-T IDLE codes as their min sqr distance is $(2\sqrt{2})^2=6$



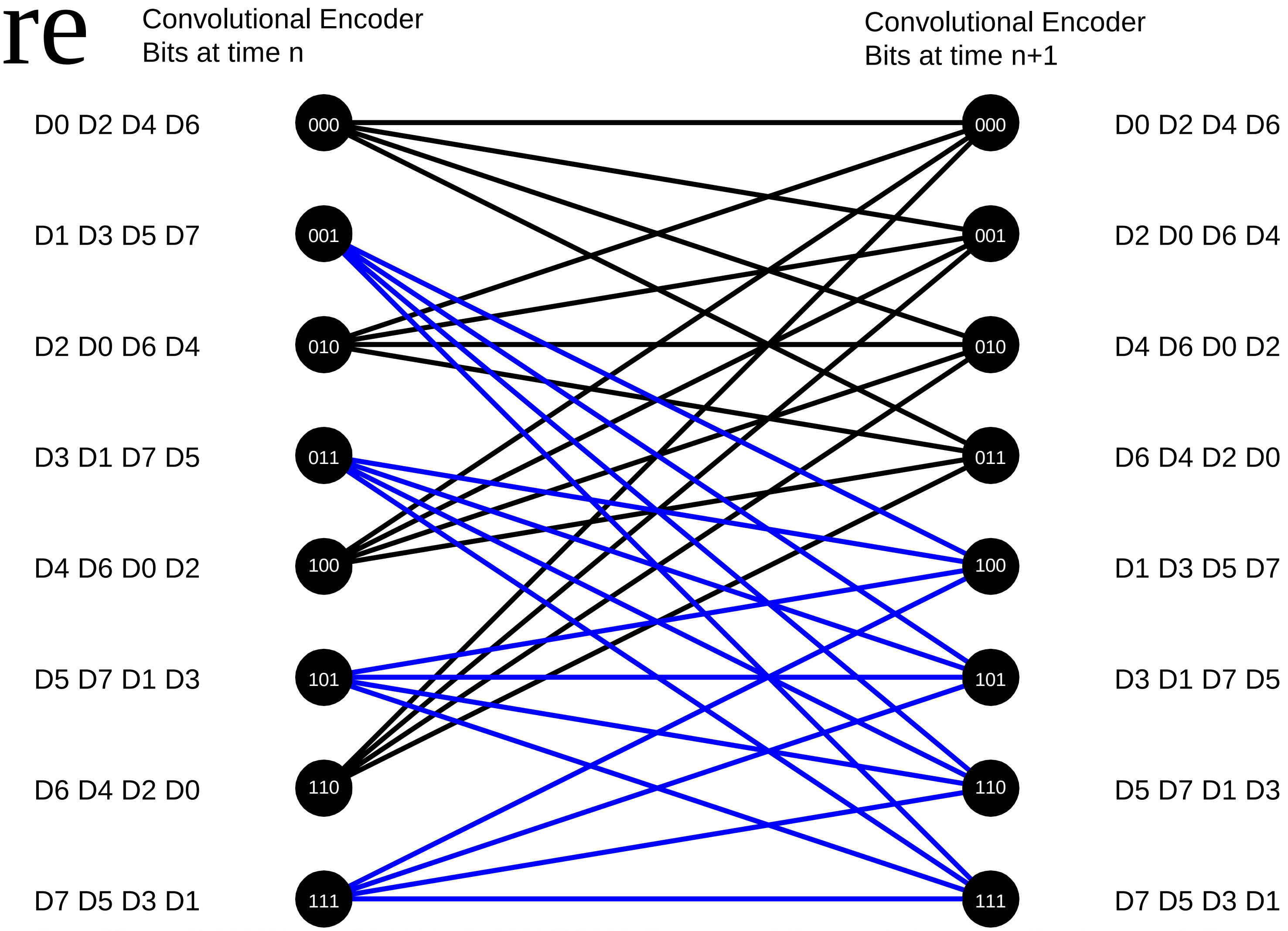
- The defined sublattice-to-symbol group mapping is used to govern the transitions in the trellis diagram.

The 1000BASE-T Convolutional Encoder



1000BASE-T Convolutional Encoder
as per IEEE Std802.3ab-1999 Section 40.3.1.3.4

The Trellis Structure



Convolutional Encoder
Bits at time n

Convolutional Encoder
Bits at time n+1

D0 D2 D4 D6	000	000	D0 D2 D4 D6
D1 D3 D5 D7	001	001	D2 D0 D6 D4
D2 D0 D6 D4	010	010	D4 D6 D0 D2
D3 D1 D7 D5	011	011	D6 D4 D2 D0
D4 D6 D0 D2	100	100	D1 D3 D5 D7
D5 D7 D1 D3	101	101	D3 D1 D7 D5
D6 D4 D2 D0	110	110	D5 D7 D1 D3
D7 D5 D3 D1	111	111	D7 D5 D3 D1

The Trellis Structure

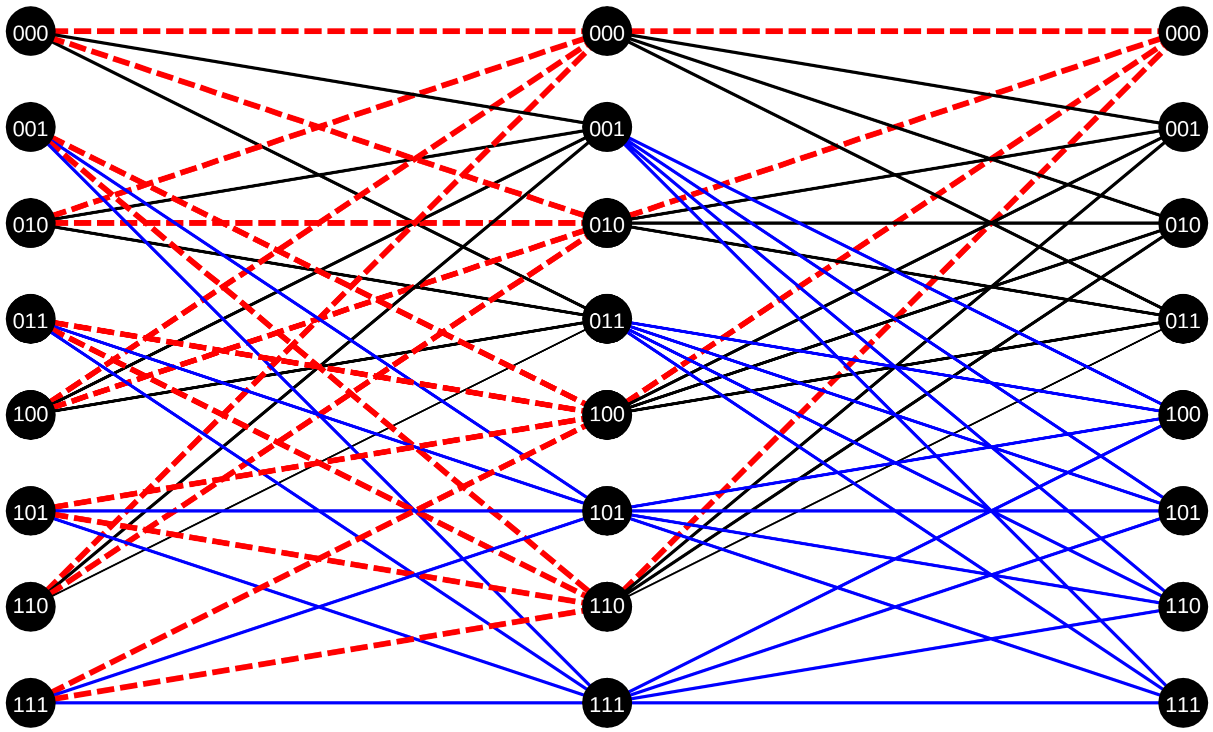
- Notice that IDLE (D0 Subset codes) does not benefit from the trellis structure - only data frames do.
- The benefit of the complexity of this system is an effective coding gain of 6dB !
- Also notice that to return to idle, two “convolutional resets” are required to guarantee that the 0 (top) state can be returned to. For example: a frame’s data may end with the trellis in state 3, the only path back to idle (state 0) requires two codes (the first to either state 4,5,6,or 7, and the next back to state 0) This is illustrated in the following slide.

Convolutional Reset (CSReset)

Convolutional Encoder
Bits at time n (end of frame data)

Convolutional Encoder
Bits at time n+1

Convolutional Encoder
Bits at time n+2



State of Encoder
at End of
Frame Data

First CSReset
in Dashed/Red: causes transition
to 000, 010, 100,or 110

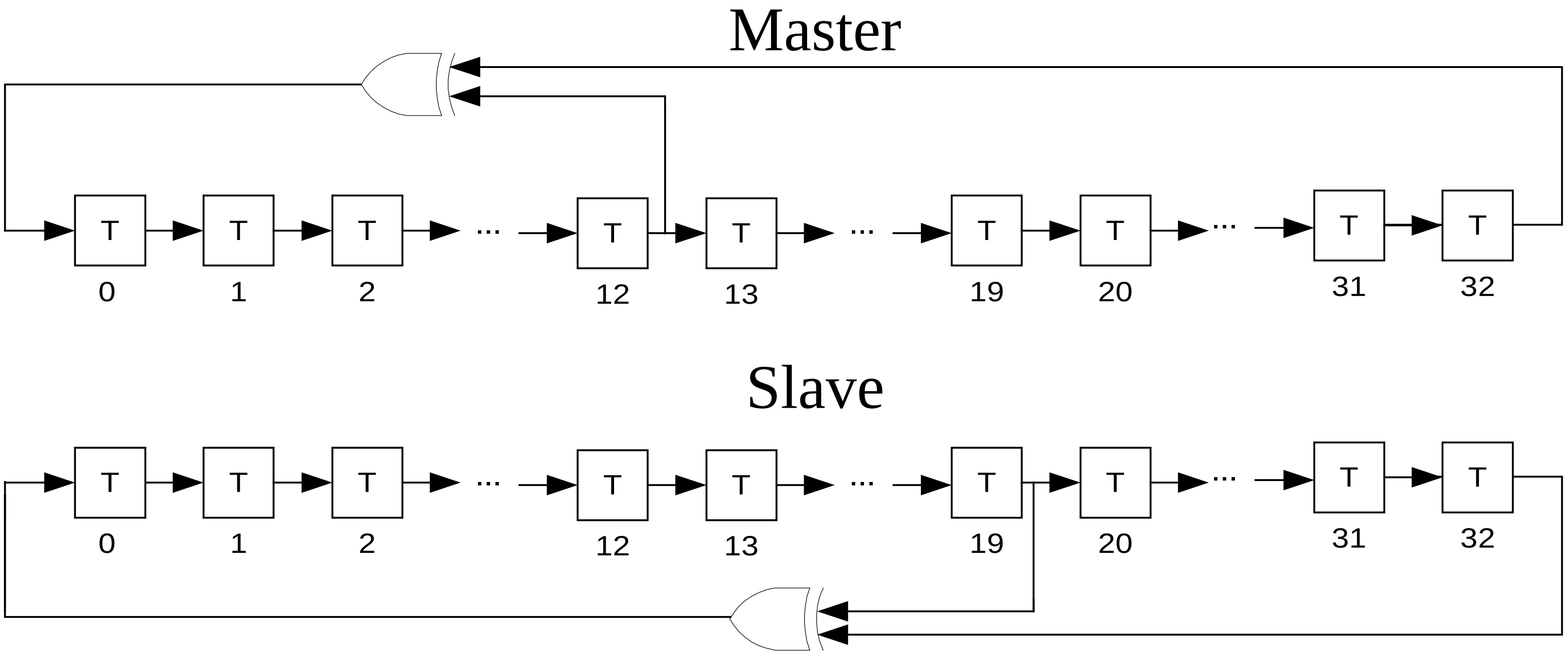
Second CSReset
in Dashed/Red: causes transition
to 000 Subset (IDLE ∈ 000)

The Master/Slave Scramblers

- The chosen scrambler (LFSR) length is 33 bits.
- That’s 2^{33}-1 or 8589934591 bits before the LFSR pattern repeats, that’s 68.72s (100Base-TX is only 2047bits)
- The two parties on a 1000BASE-T link are referred to as Master and Slave. The Master is the clock source. The Slave recovers the Master’s clock and uses that clock to transmit and receive. (How this relationship is developed is explored in Phy Control and Auto-Negotiation)

The Master/Slave Scramblers

- The LFSR Structures used are:



The Master uses the Master Side-Stream Scrambler to transmit, and the Slave Side-Stream Scrambler to receive, and vice versa for the Slave.

The 1000BASE-T Frame Format

- Constantly active: either idle, 2byte SSD, Data, or 4byte ‘end frame’

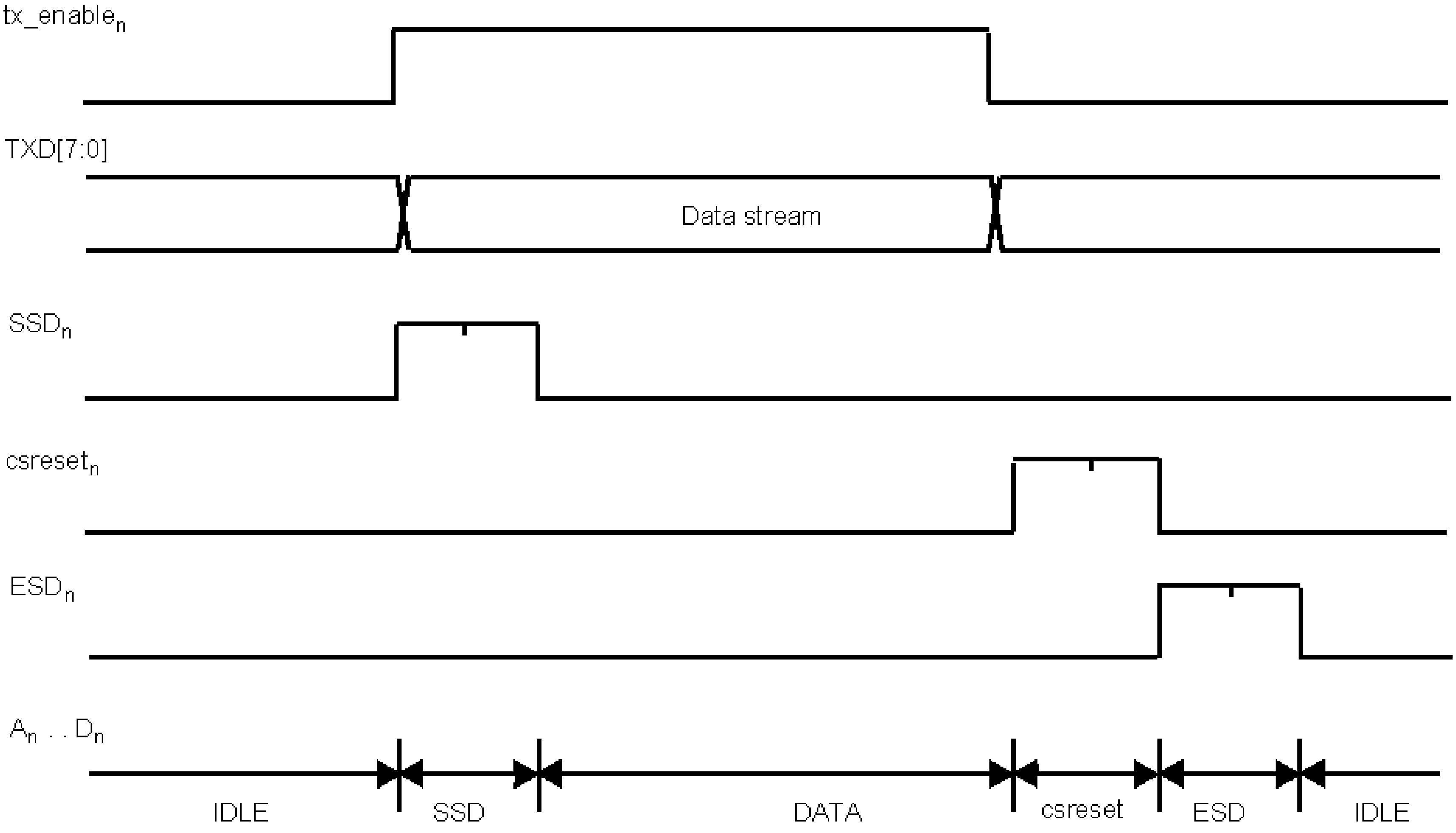


Figure 40-12—PCS sublayer to PMA timing

References

- IEEE 802.3ab/D4.2 November 15, 1998
- IEEE 802.3ab Std-1999
- Dr. Sailesh K Rao, Level One, private conversations, November 1998
- Dr. John Creigh, Broadcom, private conversations, November 1998
- “Modified 4D 8-state Trellis Coding for 1000Base-T”, by Sailesh K Rao, Level One, August 15, 1997
- “4D Encoding in Level-One’s Proposal for 1000BASE-T”, by Jaime E. Kardontchik, Advanced Micro Devices, August 1997
- “Principles of Digital Communication and Coding”, by Andrew J. Viterbi and Jim K. Omura. McGraw Hill Inc, ©1979.
- “802.3ab A Tutorial Presentation”, by Colin Mick, Chris DiMinico, Sreen Raghavan, Sailesh Rao, Mehdi Hatamian