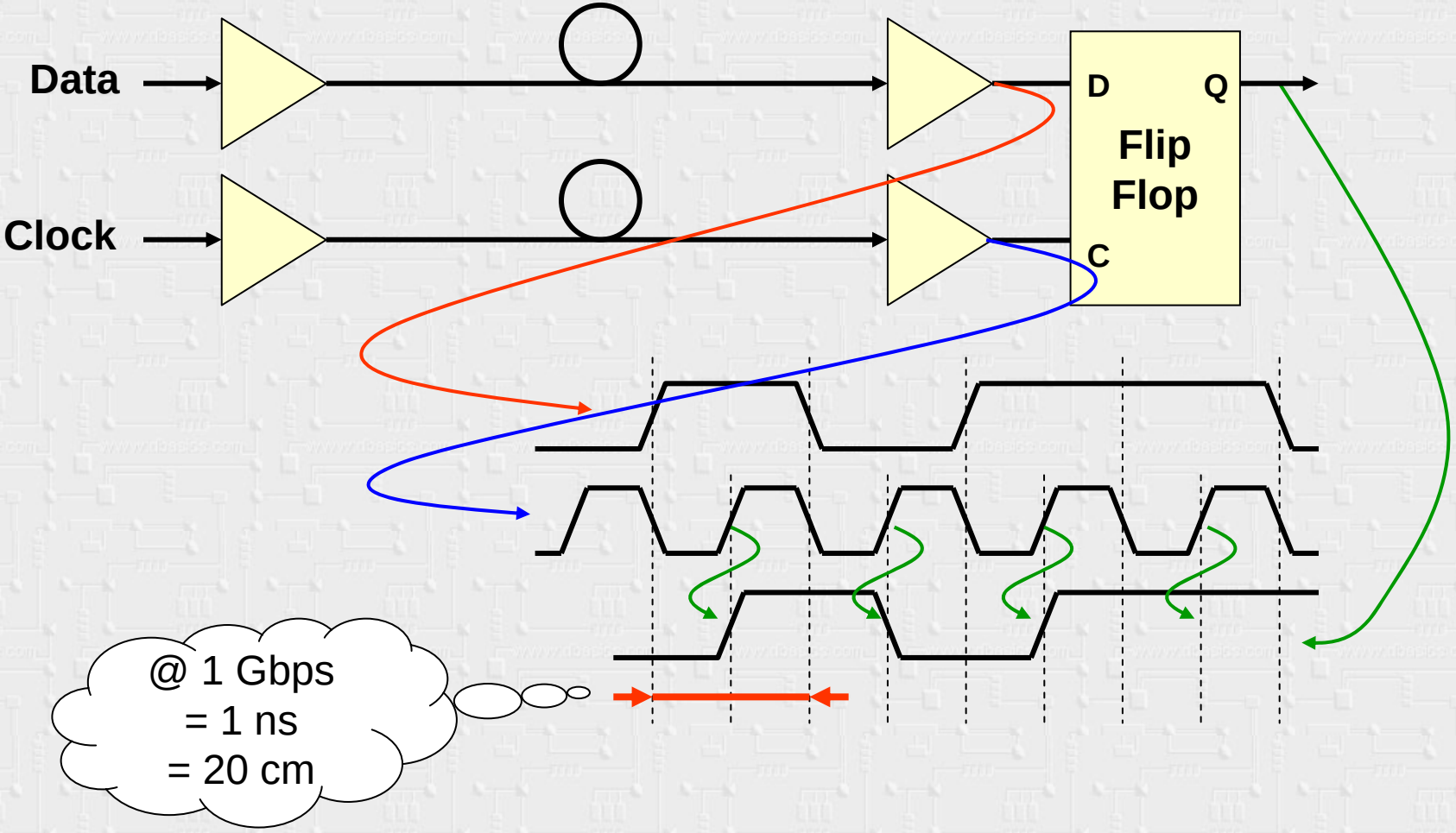
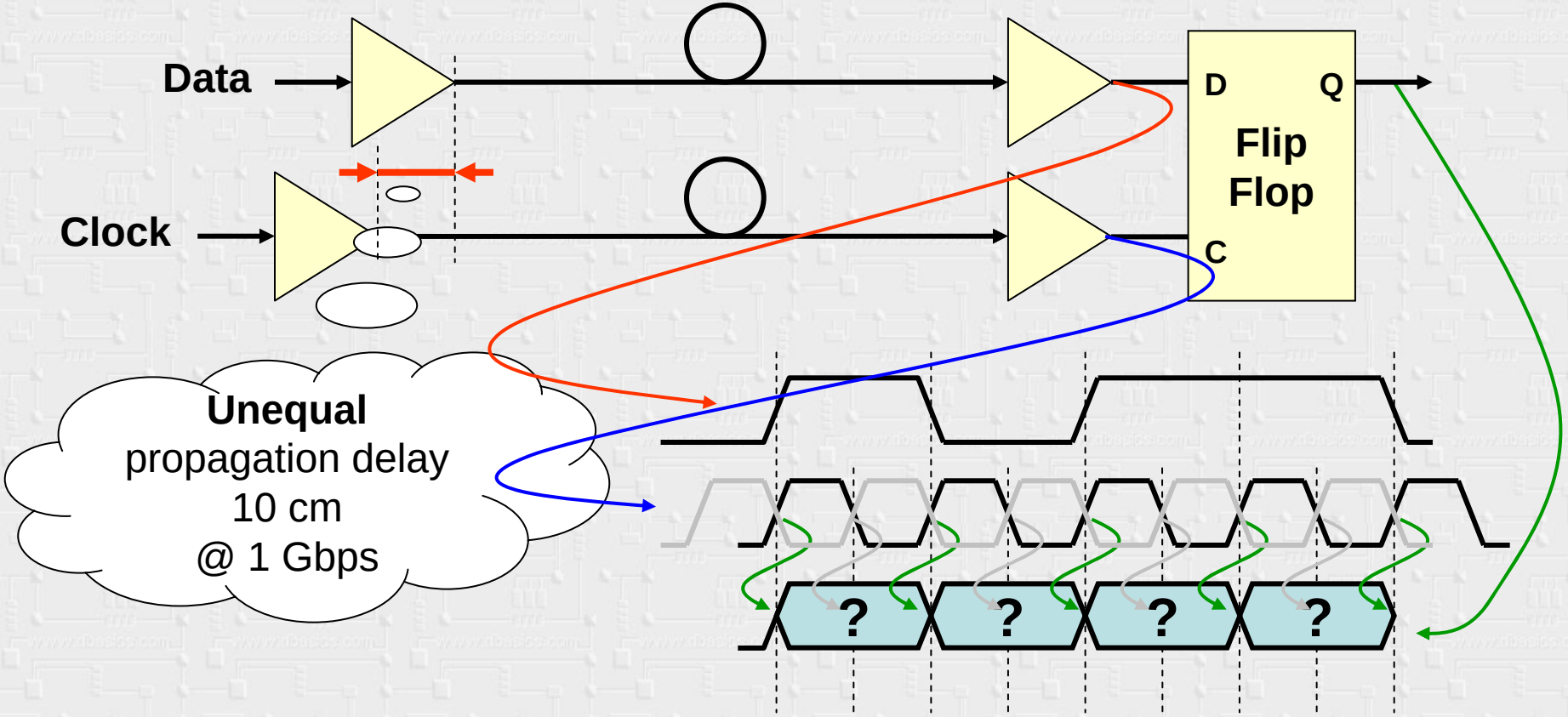
8B/10B Coding
64B/66B Coding

1. Transmission Systems
2. 8B/10B Coding
3. 64B/66B Coding
4. CIP Demonstrator Test Setup

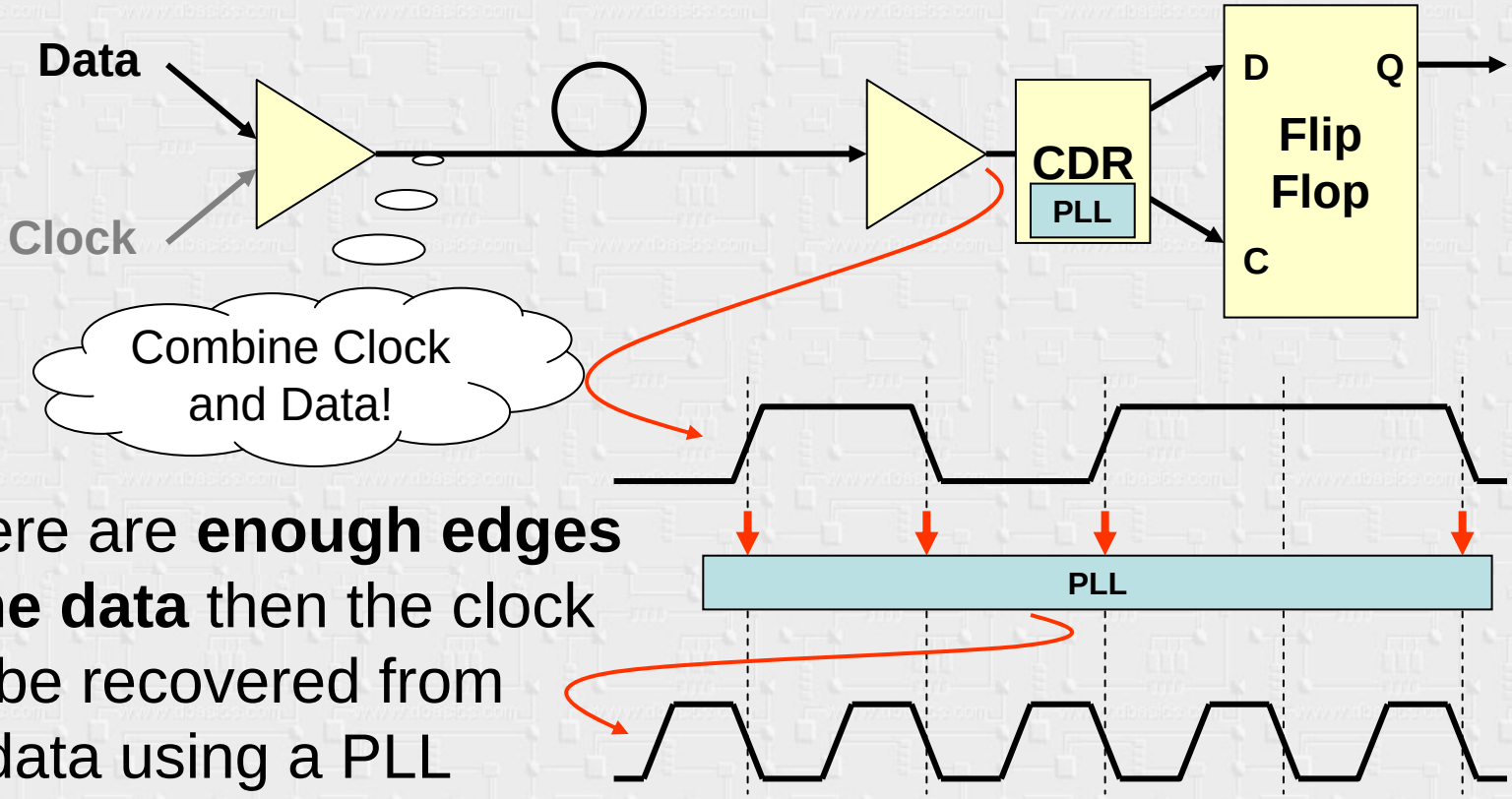
Transmission system General



Transmission system Propagation Delay



Transmission system Clock Data Recovery

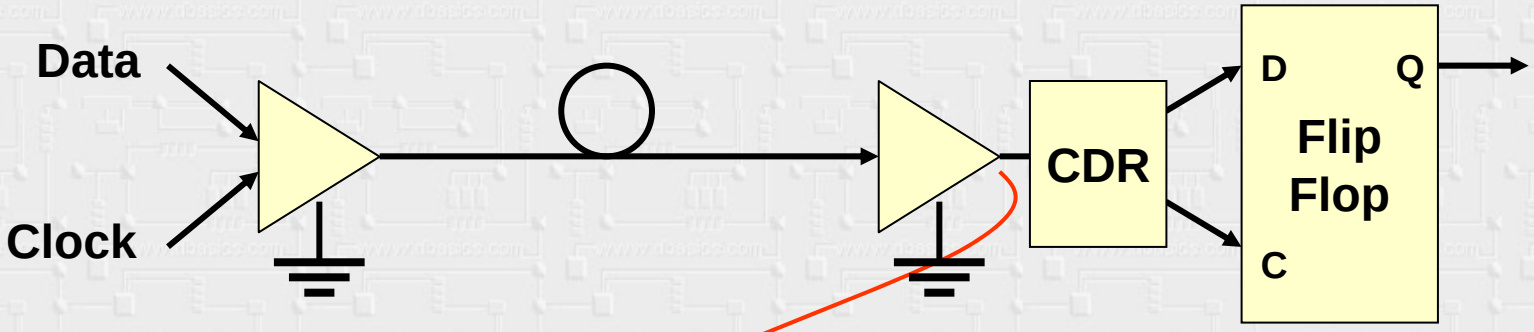


If there are **enough edges in the data** then the clock can be recovered from the data using a PLL

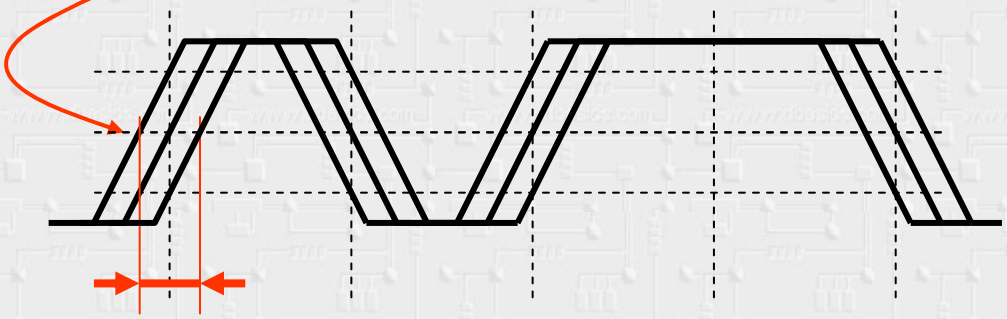
Code Properties

1. Provide enough edges in the data to enable Clock Recovery

Transmission system Receiver Threshold

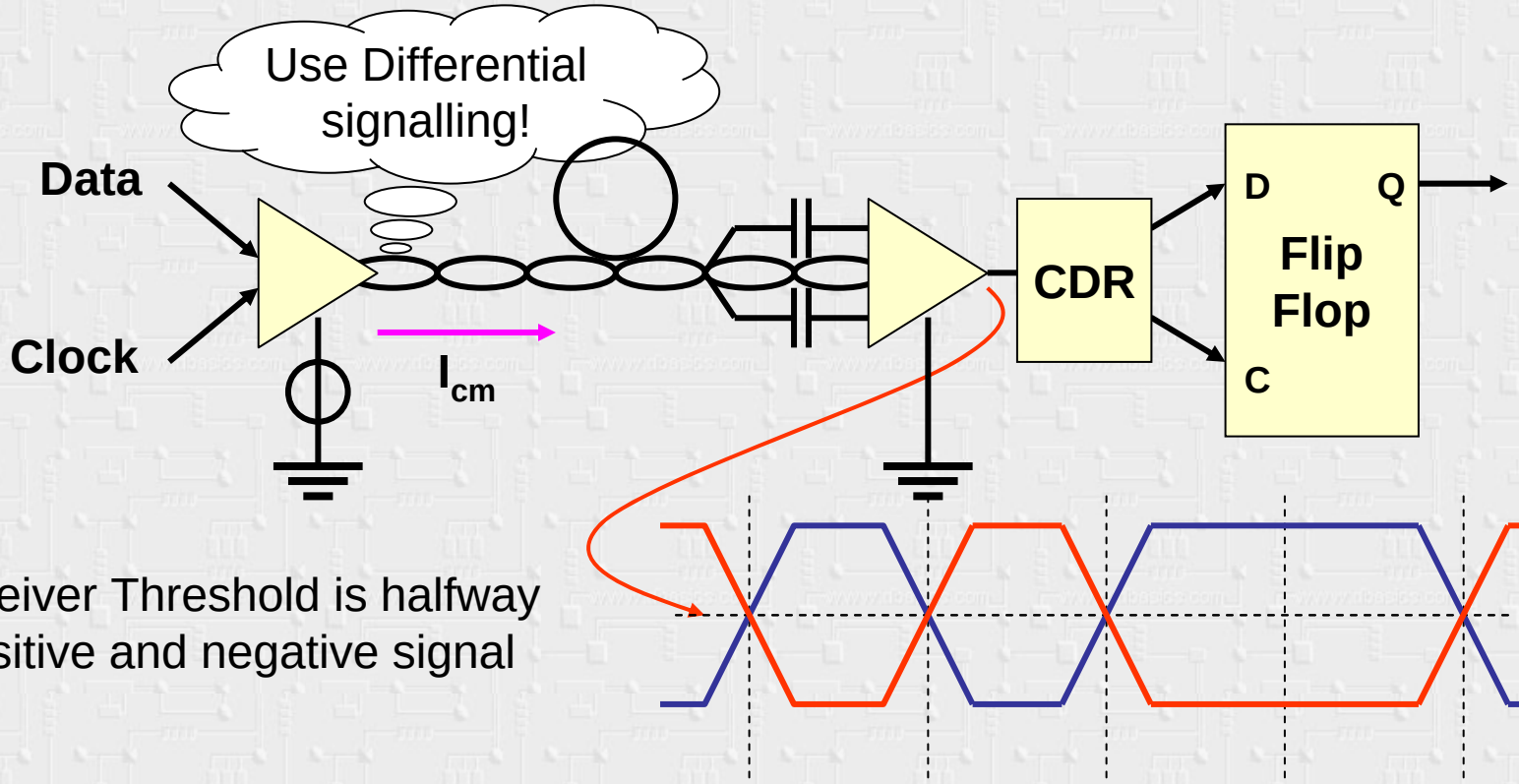


Receiver Threshold refers to "Ground" which must be the same potential as "Ground" at the transmitter!



Jitter!

Transmission



Receiver Threshold is halfway positive and negative signal

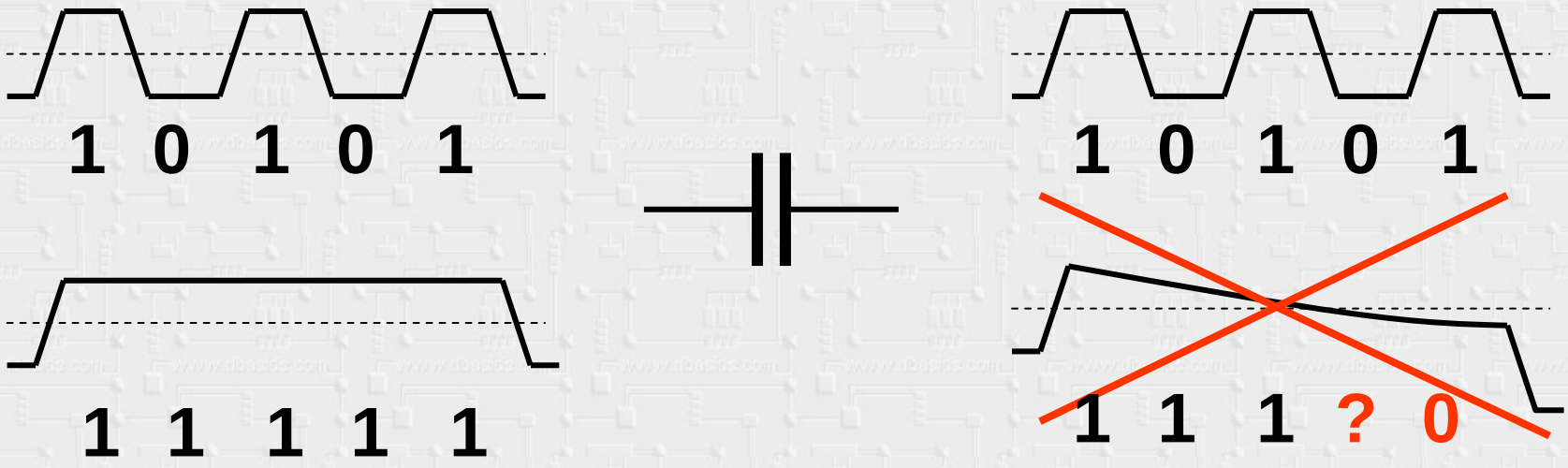
Common mode voltage differences between transmitter and receiver can result in excessive currents

Use AC Coupling

Need DC Balance

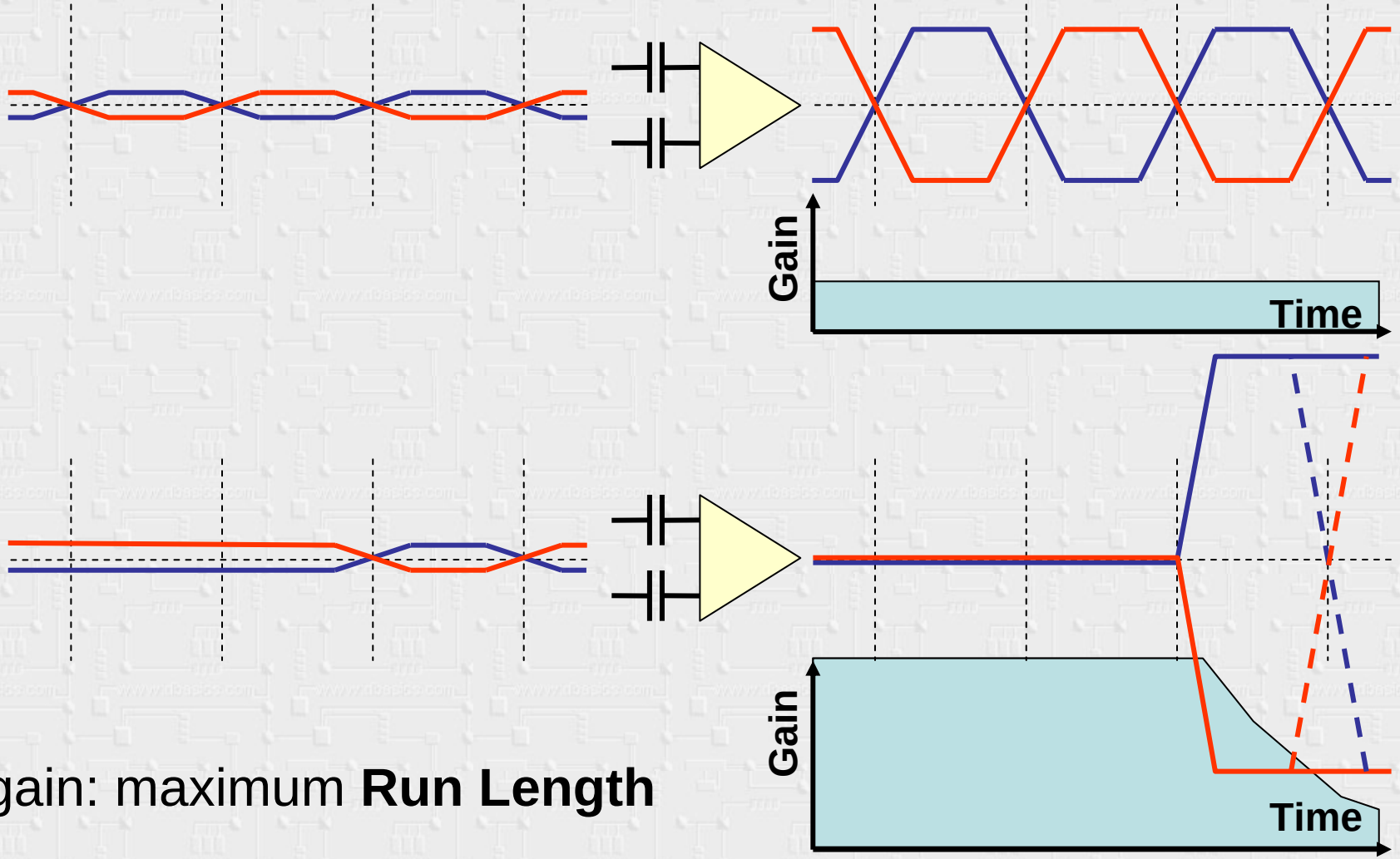
Sep 4, 2008

DC Balance



Define a maximum **Run Length**
Sent equal amount of ‘1’s and ‘0’s (Running Disparity)

Receiver Automatic Gain Control



Again: maximum **Run Length**

Code Properties

1. Provide enough edges in the data to enable Clock Recovery
2. Maximum Run Length and DC Balance

History of 8B/10B coding

A. X. Widmer
P. A. Franaszek

A DC-Balanced, Partitioned-Block, 8B/10B Transmission Code

This paper describes a byte-oriented binary transmission code and its implementation. This code is particularly well suited for high-speed local area networks and similar data links, where the information format consists of packets, variable in length, from about a dozen up to several hundred 8-bit bytes. The proposed transmission code translates each source byte into a constrained 10-bit binary sequence which has excellent performance parameters near the theoretical limits for 8B/10B codes. The maximum run length is 5 and the maximum digital sum variation is 6. A single error in the encoded bits can, at most, generate an error burst of length 5 in the decoded domain. A very simple implementation of the code has been accomplished by partitioning the coder into 5B/6B and 3B/4B subordinate coders.

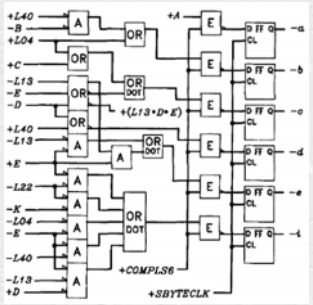
IBM J. RES. DEVELOP. • VOL. 27 • NO. 5 • SEPTEMBER 1983

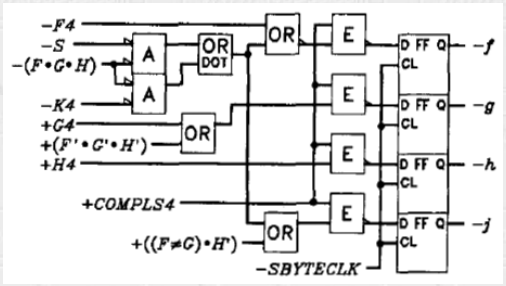
8B/10B coding

- Maximum run length = 5
- Running Disparity => DC Balance

- 8 bits => 256 different values
- 10 bits => 1024 different values
- 8B10B code is built out of:

  (5 to 6 bit code) + (3 to 4 bit code)



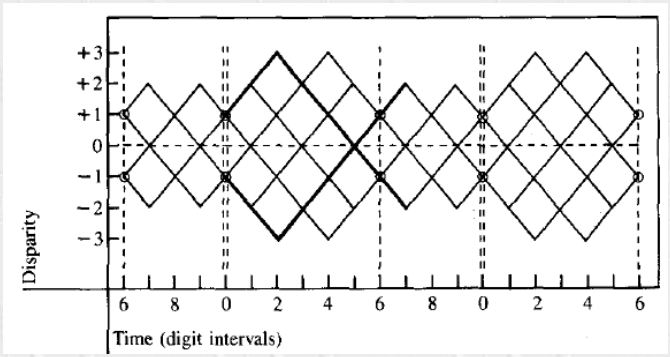


8B/10B coding

- Not all 1024 values are useful.
  - For example "**0100000011**" has run length 6
- For most of the 256 (8B) values a positive and a negative 10B value is selected depending on the "Current Running Disparity"
  - For example:

D7.0 = 1110001011 (Current RD-)

D7.0 = 0001110100 (Current RD+)



- So about 512 useful values are selected from 1024…
- There are still a few codes left!

8B/10B coding table

- 256 “Data” Characters:

Table C-1: **Valid Data Characters**

Data Byte Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
D0.0	000 00000	100111 0100	011000 1011
D1.0	000 00001	011101 0100	100010 1011
D2.0	000 00010	101101 0100	010010 1011
D3.0	000 00011	110001 1011	110001 0100
D4.0	000 00100	110101 0100	001010 1011
D5.0	000 00101	101001 1011	101001 0100
D6.0	000 00110	011001 1011	011001 0100
D7.0	000 00111	111000 1011	000111 0100
D8.0	000 01000	111001 0100	000110 1011
D9.0	000 01001	100101 1011	100101 0100
⋮			
D31.7	111 11111	101011 0001	010100 1110

8B/10B coding table

- There were still a few codes left!
- 12 “Special” K Characters:

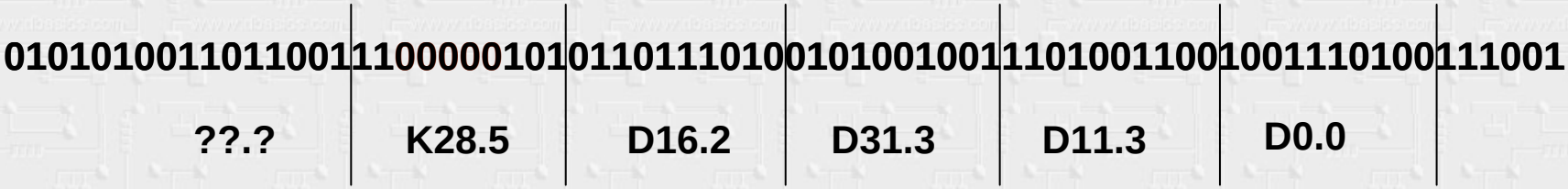
Comma Characters

“The only patterns that have 5 consecutive ‘1’s or ‘0’s

Table C-2: **Valid Control K Characters**

Special Code Name	Bits HGF EDCBA	Current RD – abcdei fghj	Current RD + abcdei fghj
K28.0	000 11100	001111 0100	110000 1011
K28.1	001 11100	001111 1001	110000 0110
K28.2	010 11100	001111 0101	110000 1010
K28.3	011 11100	001111 0011	110000 1100
K28.4	100 11100	001111 0010	110000 1101
K28.5	101 11100	001111 1010	110000 0101
K28.6	110 11100	001111 0110	110000 1001
K28.7 (1)	111 11100	001111 1000	110000 0111
K23.7	111 10111	111010 1000	000101 0111
K27.7	111 11011	110110 1000	001001 0111
K29.7	111 11101	101110 1000	010001 0111
K30.7	111 11110	011110 1000	100001 0111

8B/10B code “K” Characters



- Comma characters K28.1/K28.5/K28.7 are used *word* alignment
- Create “ordered sets”
  - For example Fibre Channel **S**tart **O**f **F**rame (SOF) = K28.5/D21.5/D23.0/D23.0
  - K30.7 = Error Propagate
  - K28.3 = Carrier Extend

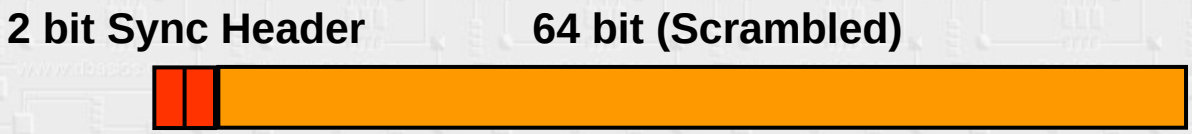
8B/10B code properties

1. Provide enough edges in the data to enable Clock Recovery
2. Maximum Run Length and DC Balance
3. Add special characters
   - Comma for *word* alignment
   - Control characters

64B/66B Coding

64B/66B
(IEEE std 802.3ae-2002, Clause 49)

- 20% of the bandwidth for 8B/10B is overhead
- For 64B/66B overhead is 3%



2 bit Sync Header

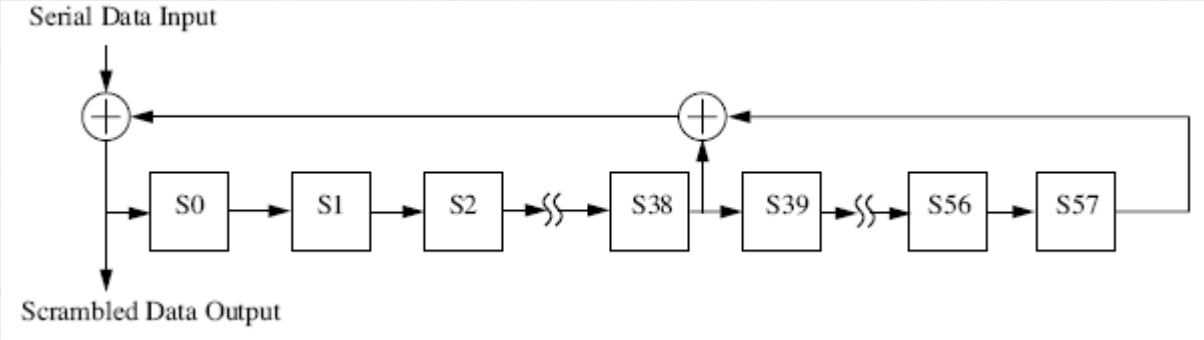
00	Code Error
01	64 bit = **Data** field
10	64 bit = Mixed **Data / Control** field
11	Code Error

- Once every 66 bits there is at least a "01" or a "10" transition

  => Receiver Block Sync (it may take a while…)

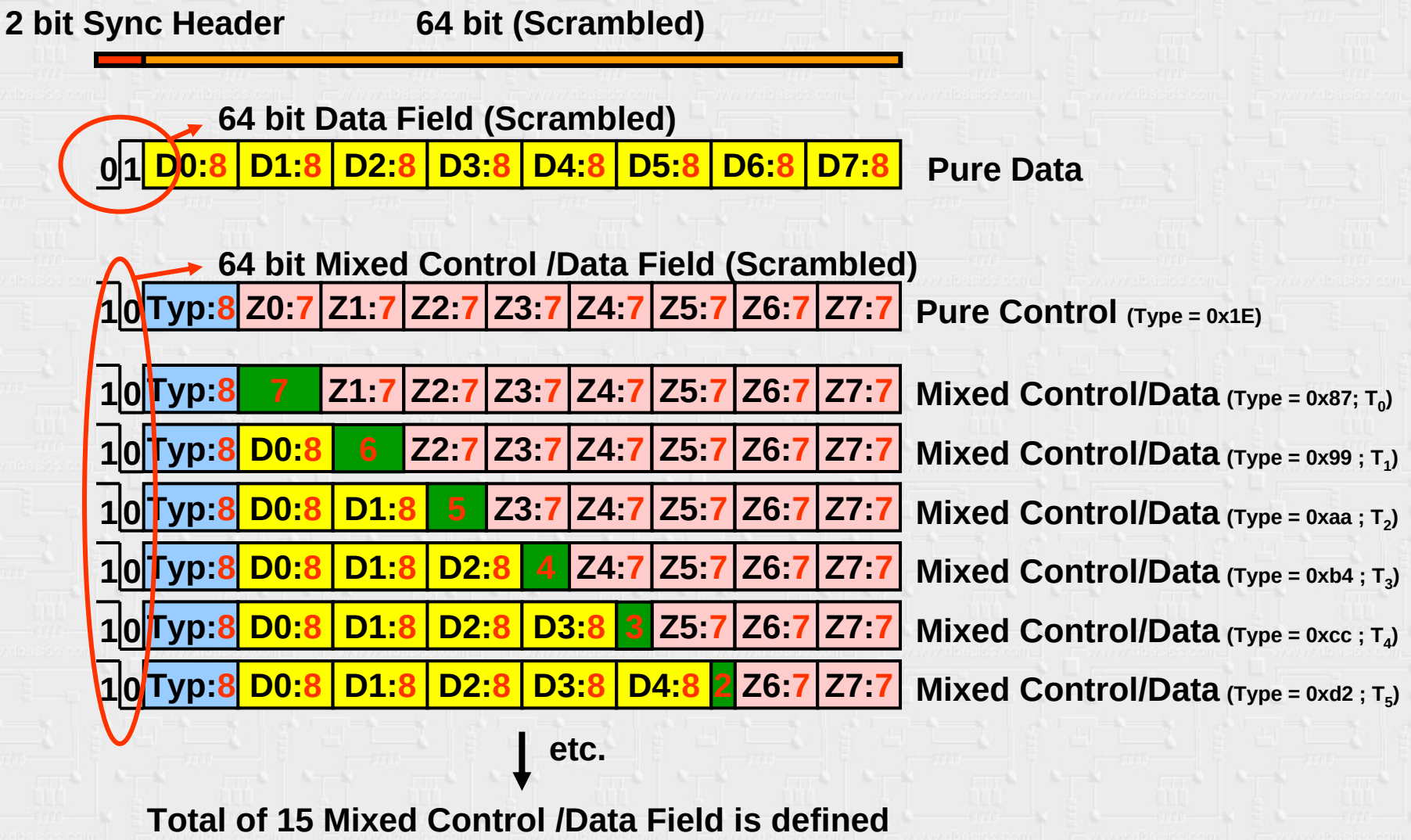
64B/66B Scrambler

- Scrambler: $X^{58}+X^{39}+1$



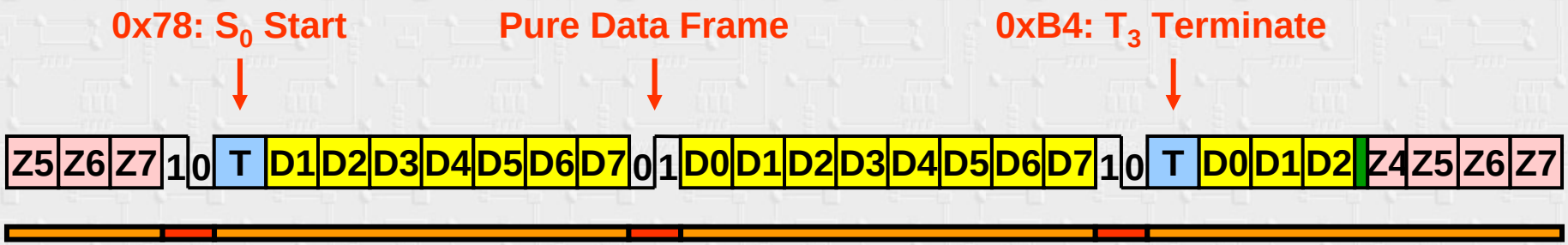
- Tries to mix up the data to avoid long sequences of ‘1’s and ‘0’s
- Note: there is no guarantee that on a bad day the output of the scrambler produces all ‘0’s or all ‘1’s

64B/66B

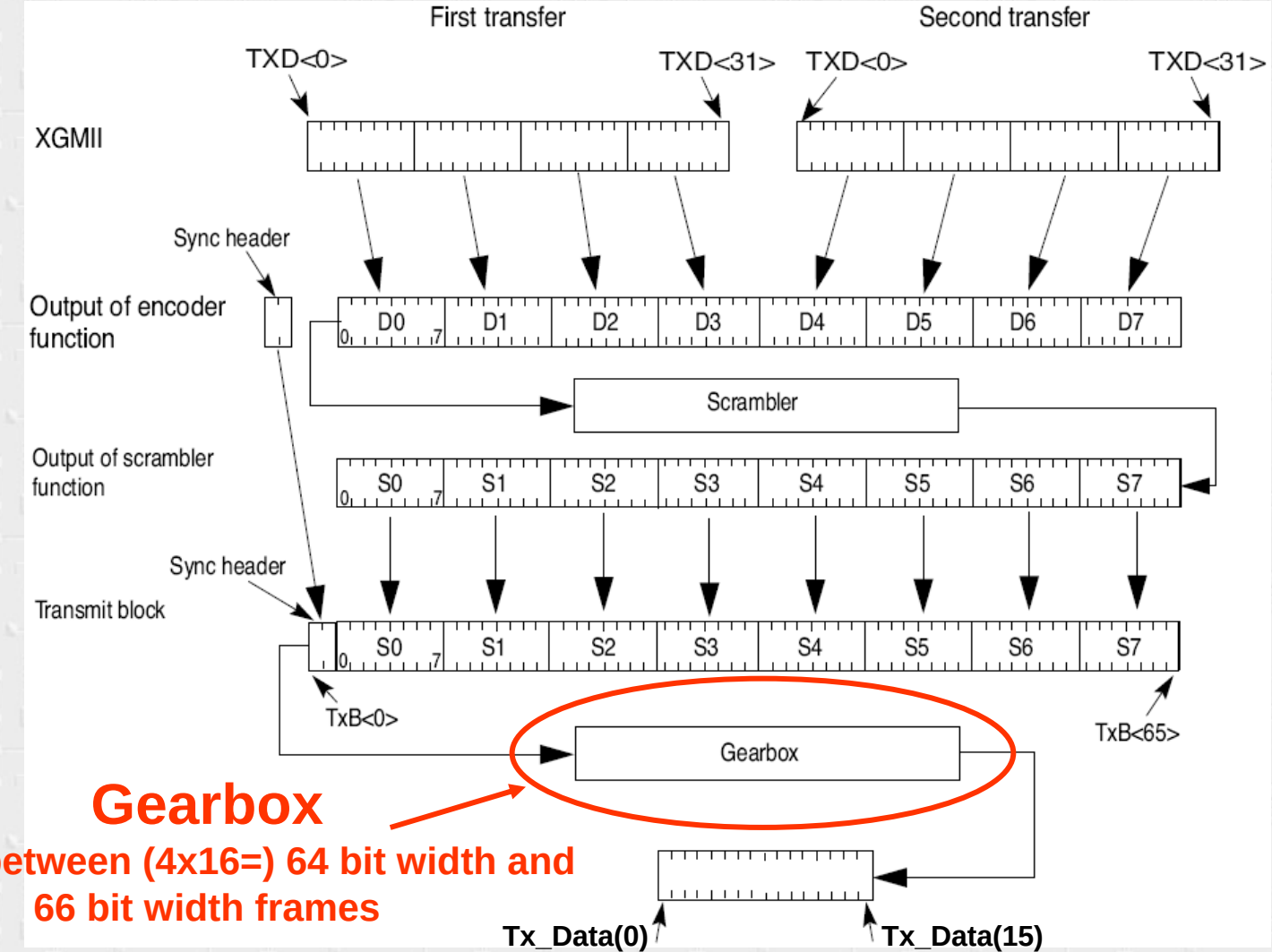

2 bit Sync Header
64 bit (Scrambled)
64 bit Data Field (Scrambled)
0 1 D0:8 D1:8 D2:8 D3:8 D4:8 D5:8 D6:8 D7:8
Pure Data
64 bit Mixed Control /Data Field (Scrambled)
1 0 Typ:8 Z0:7 Z1:7 Z2:7 Z3:7 Z4:7 Z5:7 Z6:7 Z7:7
Pure Control (Type = 0x1E)
1 0 Typ:8 7 Z1:7 Z2:7 Z3:7 Z4:7 Z5:7 Z6:7 Z7:7
Mixed Control/Data (Type = 0x87; T0)
1 0 Typ:8 D0:8 6 Z2:7 Z3:7 Z4:7 Z5:7 Z6:7 Z7:7
Mixed Control/Data (Type = 0x99 ; T1)
1 0 Typ:8 D0:8 D1:8 5 Z3:7 Z4:7 Z5:7 Z6:7 Z7:7
Mixed Control/Data (Type = 0xaa ; T2)
1 0 Typ:8 D0:8 D1:8 D2:8 4 Z4:7 Z5:7 Z6:7 Z7:7
Mixed Control/Data (Type = 0xb4 ; T3)
1 0 Typ:8 D0:8 D1:8 D2:8 D3:8 3 Z5:7 Z6:7 Z7:7
Mixed Control/Data (Type = 0xcc ; T4)
1 0 Typ:8 D0:8 D1:8 D2:8 D3:8 D4:8 2 Z6:7 Z7:7
Mixed Control/Data (Type = 0xd2 ; T5)
etc.
Total of 15 Mixed Control /Data Field is defined

64B/66B Example

- Transmitting a sequence of 18 bytes could look like:



64B/66B Physical Coding Sub layer (PCS) Transmit bit ordering



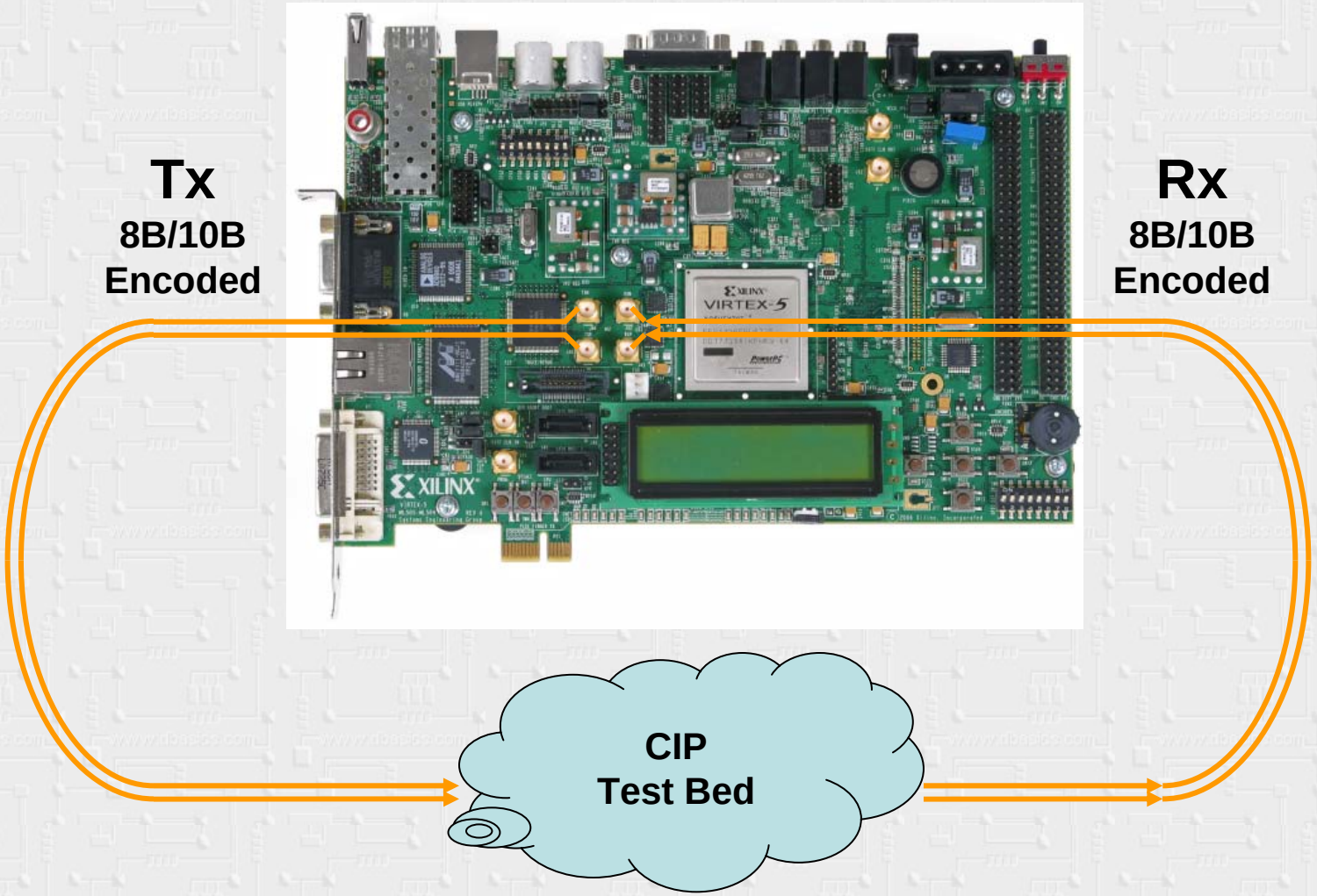
Gearbox

Adapts between (4x16=) 64 bit width and 66 bit width frames

Comparison

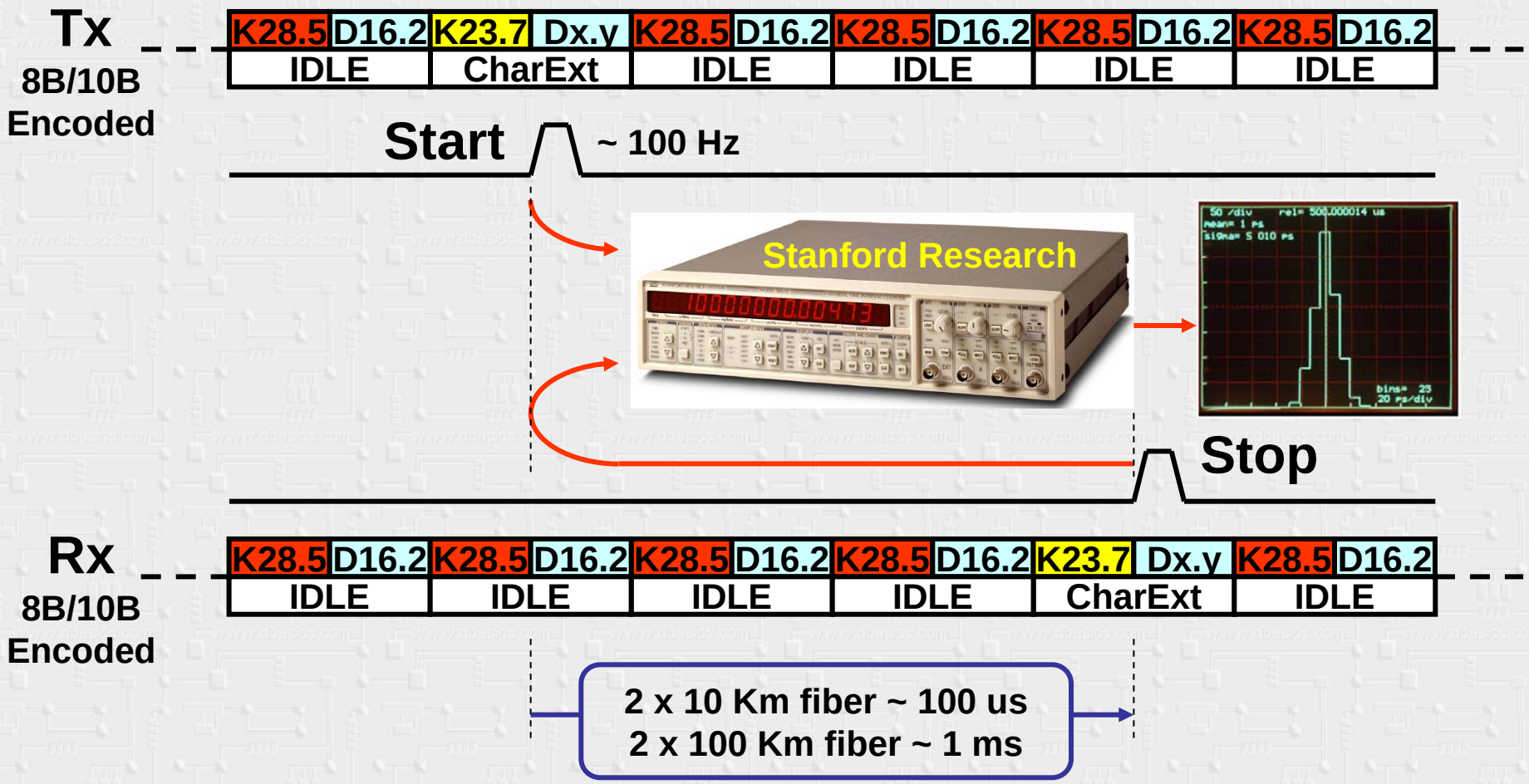
	8B/10B	64B/66B
Run Length	5	Relies on Scrambler
DC Balance	Excellent	Not guaranteed Demanding for receiver
Bit Synchronization Clock Recovery	Excellent	Relies on Scrambler, but al least one transition per 66 bits
Word Synchronization	"Comma" K-Characters	Sync-Header
Control Characters	K-Characters	Control-Codes

CIP Demonstrator Test Setup



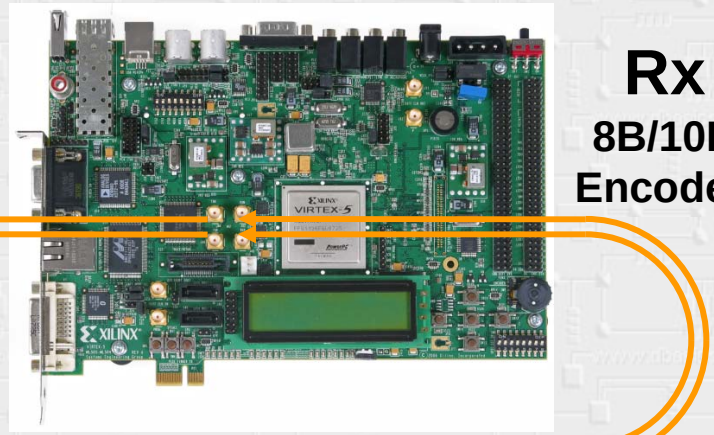
Delay…?

CIP Demonstrator Test Setup

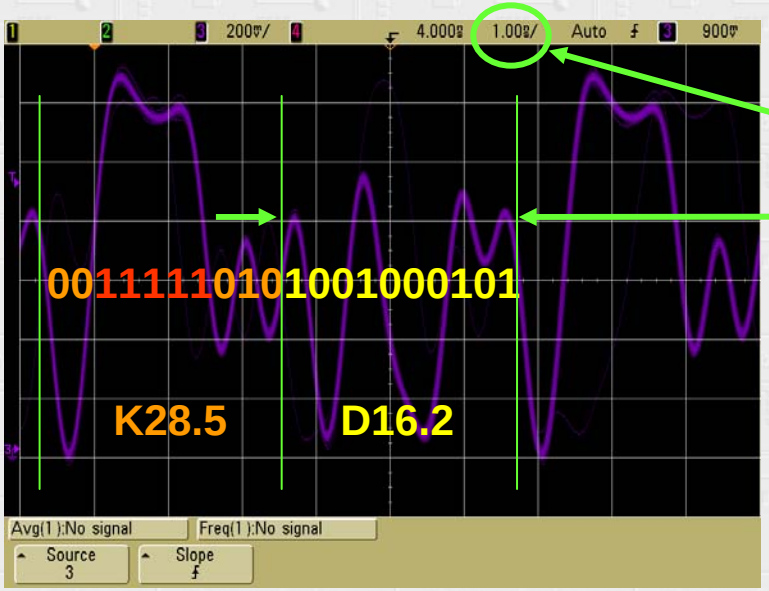


First Results

Tx
8B/10B Encoded



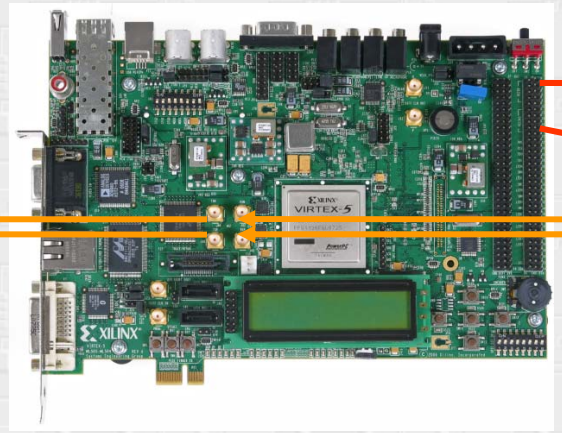
Rx
8B/10B Encoded



1 ns/div
10 bits = 3.2 ns

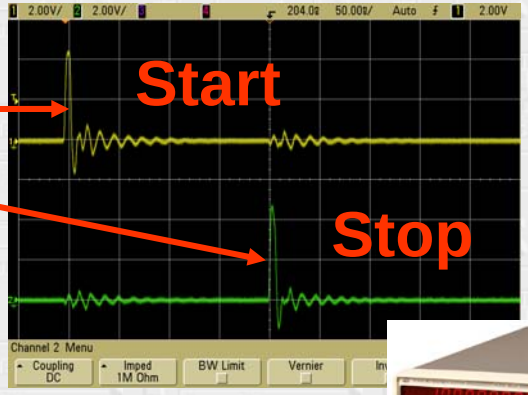
3.125 Gbps Measured With a 1.5 GHz active probe (Agilent 1156A)

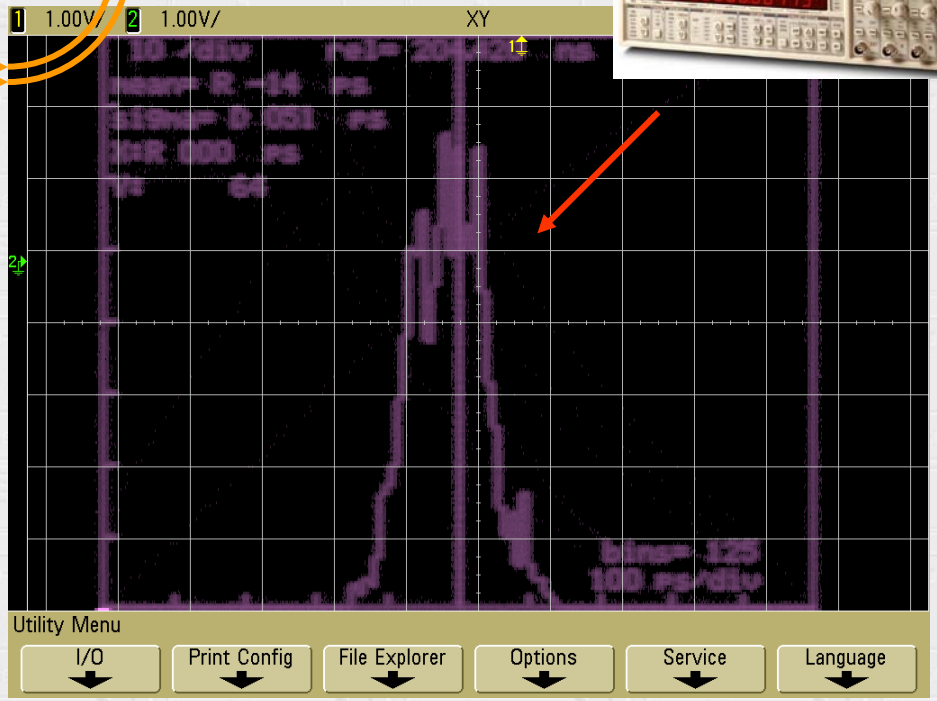
First Results



Tx
8B/10B Encoded

Rx
8B/10B Encoded





Mean: 204.420 ps
Jitter: ~ 50 ps