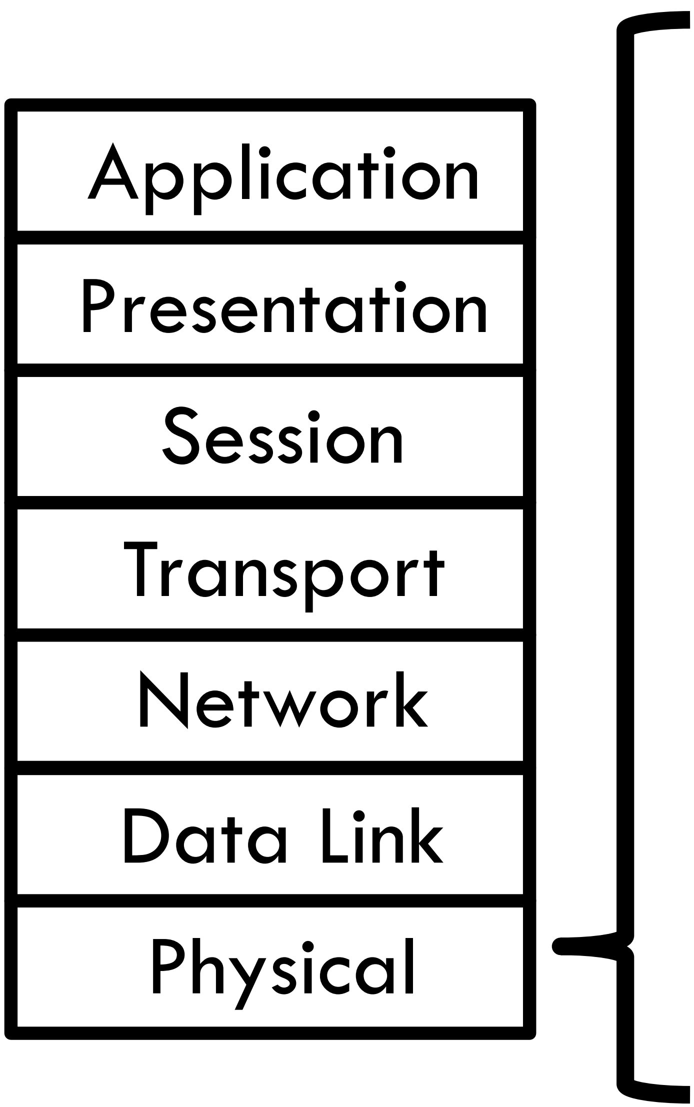
CSCI-351
Data communication and Networks

Lecture 5: Physical Layer
(The layer for EE majors…)

The slide is built with the help of Prof. Alan Mislove, Christo Wilson, and David Choffnes's class

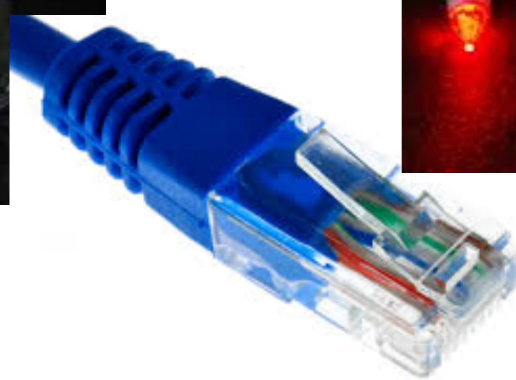
Physical Layer



- Function:
  - Get bits across a physical medium
- Key challenge:
  - How to represent bits in analog
  - Ideally, want high-bit rate
  - But, must avoid desynchronization

Key challenge

- Digital computers
  - 0s and 1s
- Analog world
  - Amplitudes and frequencies

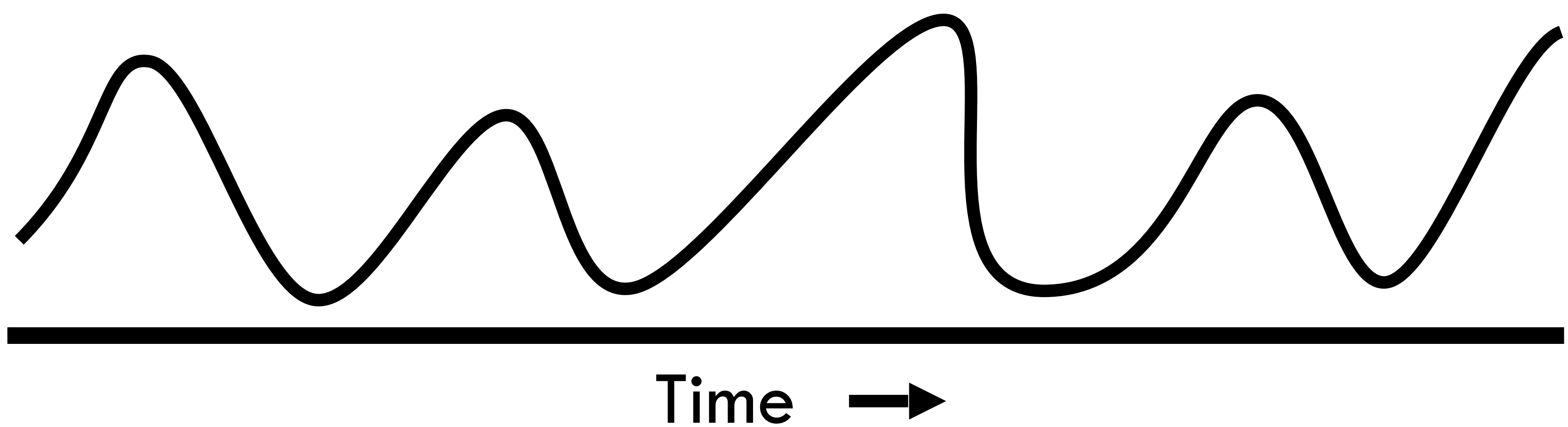


Assumptions

- We have two discrete signals, high and low, to encode 1 and 0

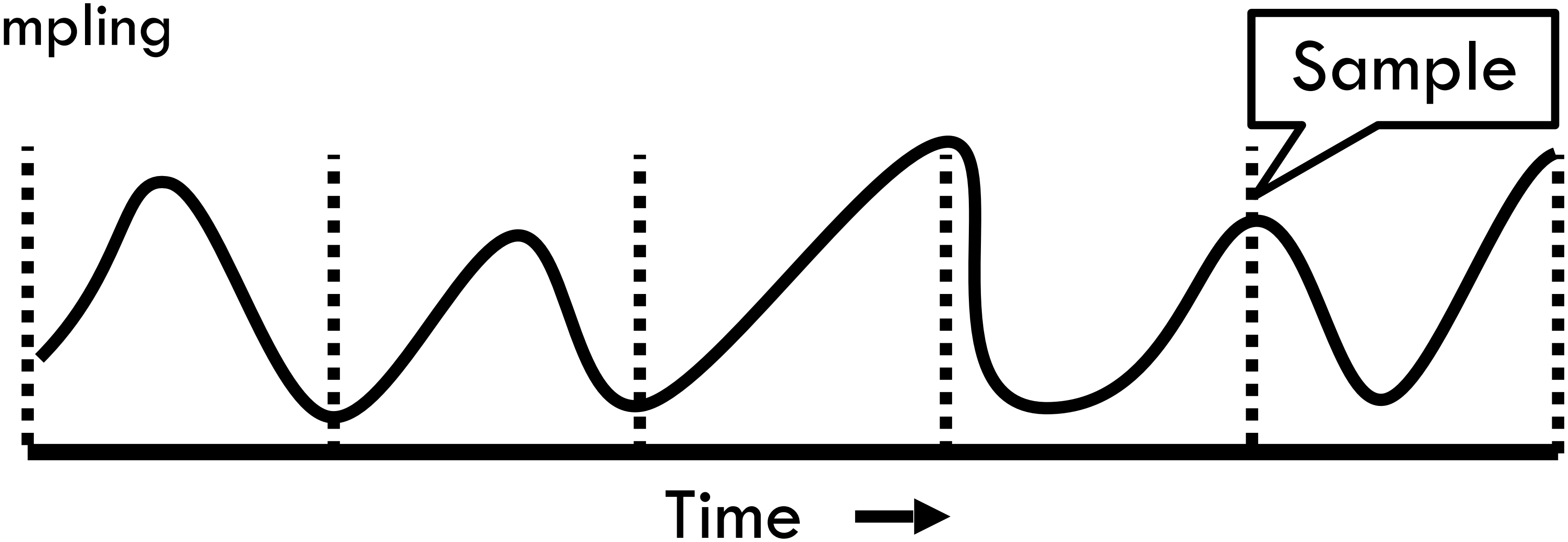
Assumptions

- We have two discrete signals, high and low, to encode 1 and 0
- Transmission is synchronous, i.e. there is a clock that controls signal sampling



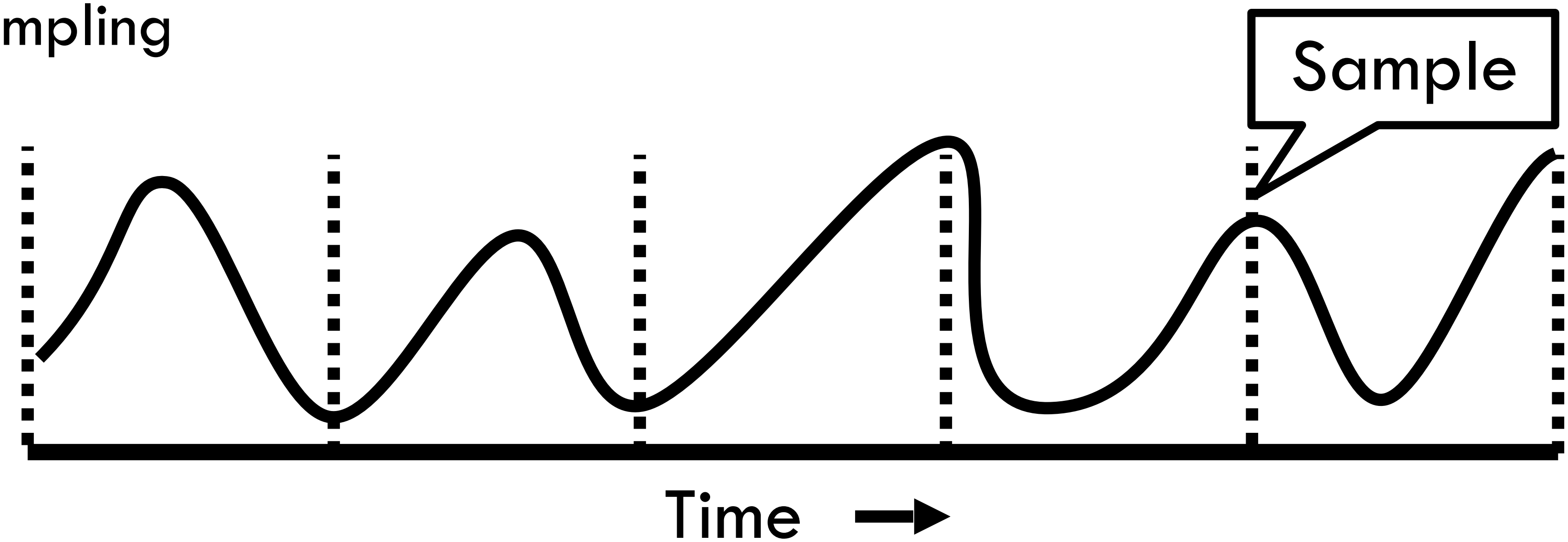
Assumptions

- We have two discrete signals, high and low, to encode 1 and 0
- Transmission is synchronous, i.e. there is a clock that controls signal sampling

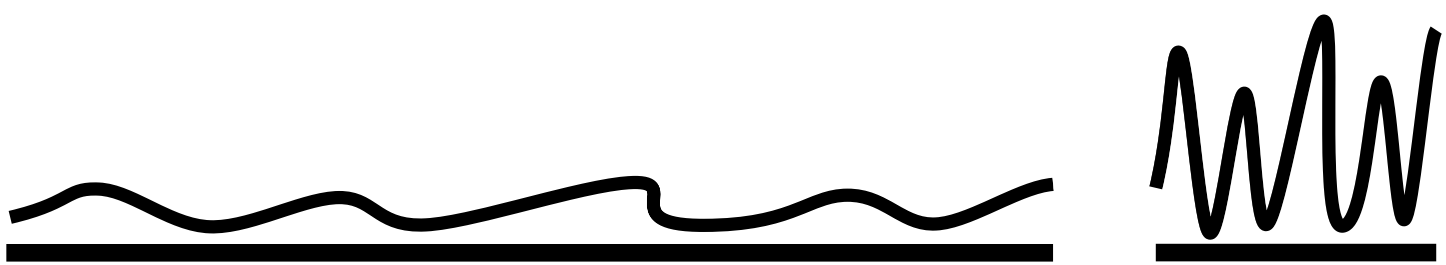


Assumptions

- We have two discrete signals, high and low, to encode 1 and 0
- Transmission is synchronous, i.e. there is a clock that controls signal sampling



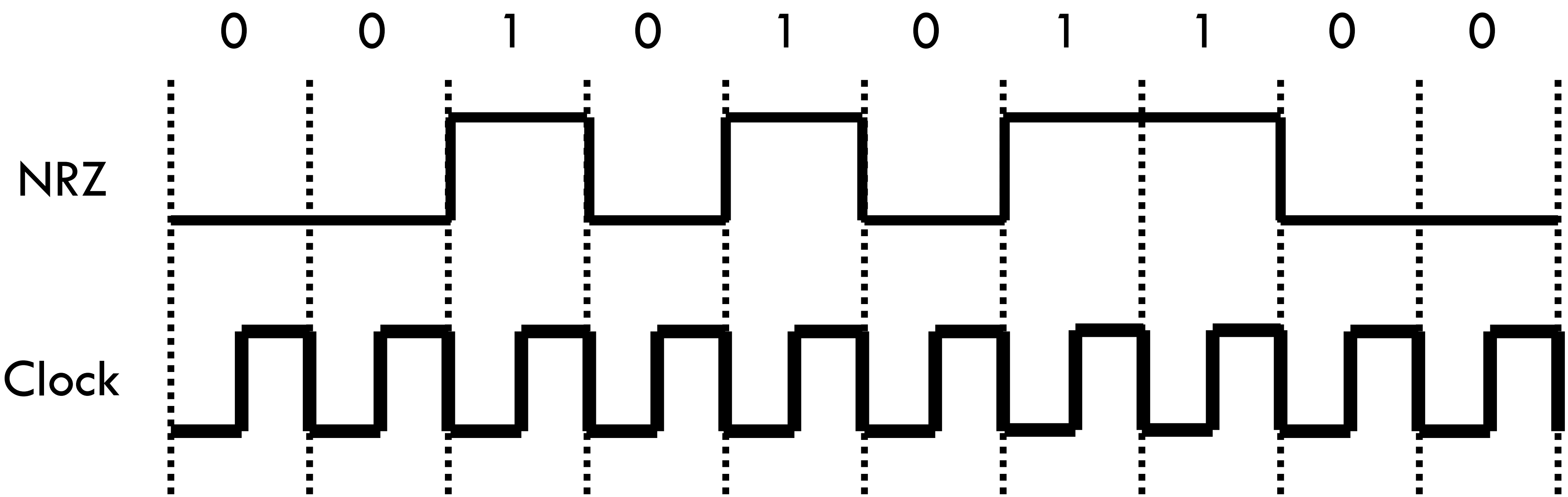
- Amplitude and duration of signal must be significant



Non-Return to Zero (NRZ)

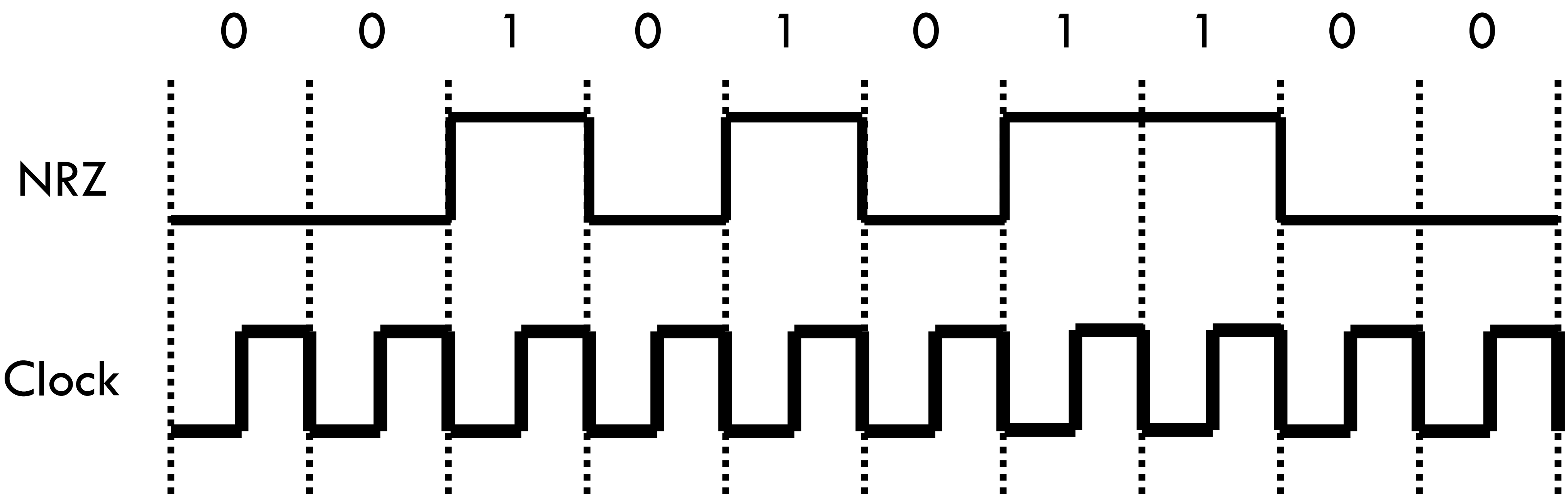
5

- 1 → high signal, 0 → low signal



Non-Return to Zero (NRZ)

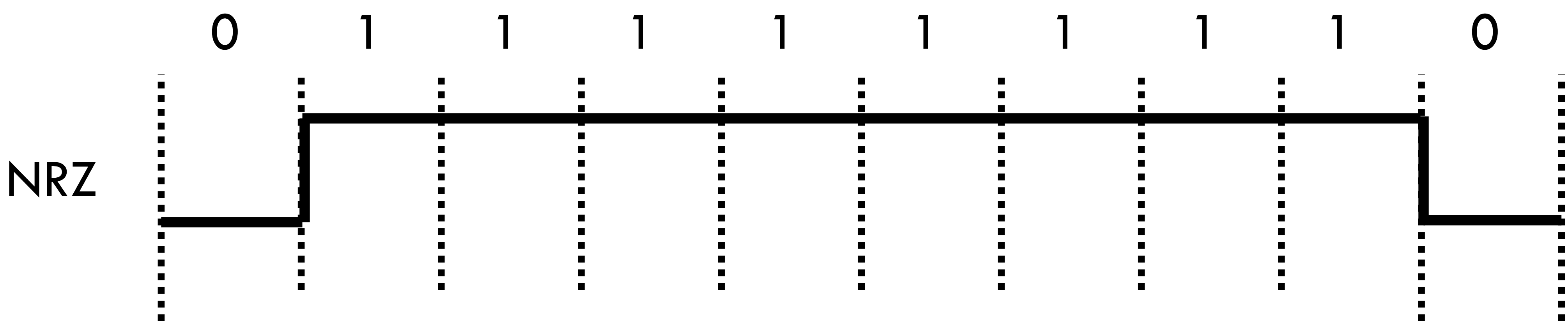
- 1 → high signal, 0 → low signal



- Problem: long strings of 0 or 1 cause desynchronization
  - How to distinguish lots of 0s from no signal?
  - How to recover the clock during lots of 1s?

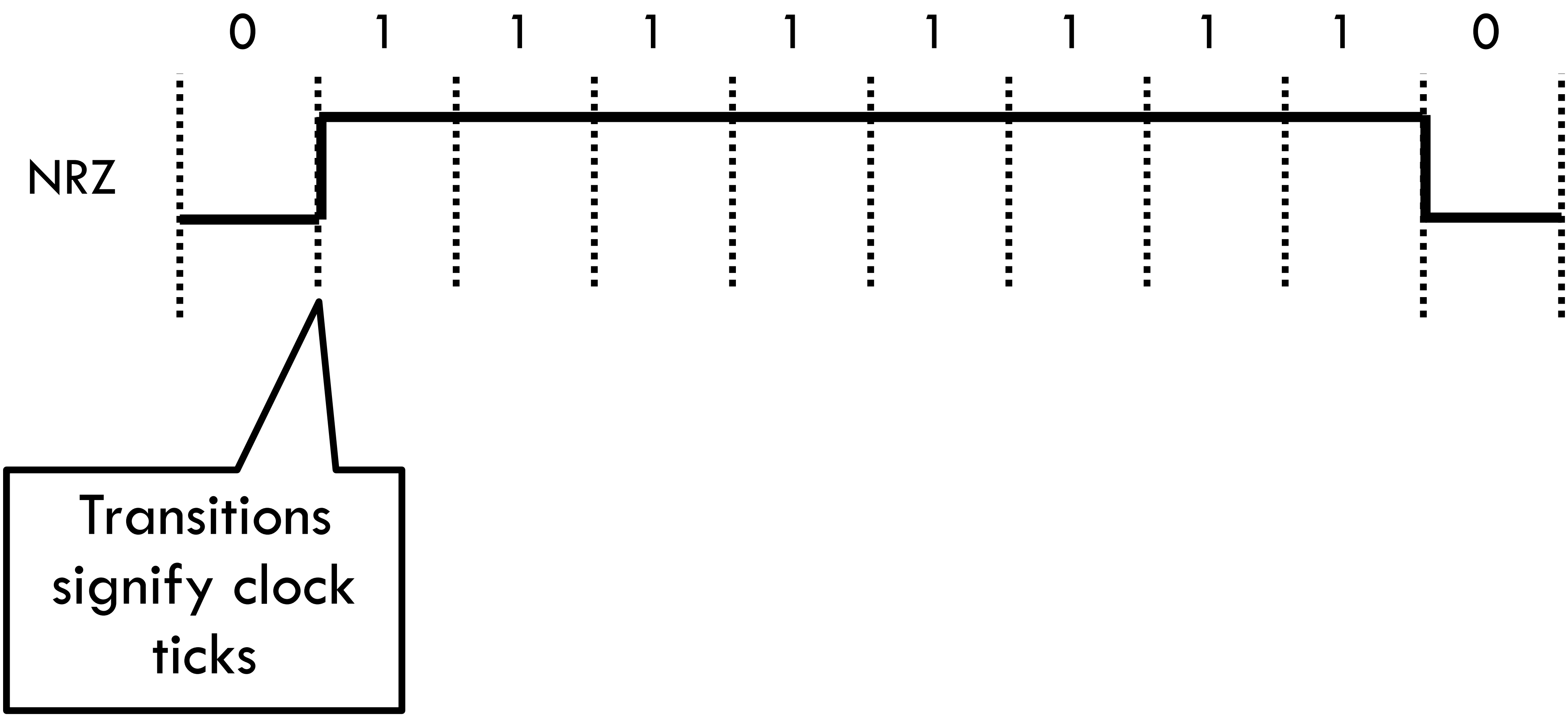
Desynchronization

- Problem: how to recover the clock during sequences of 0’s or 1’s?



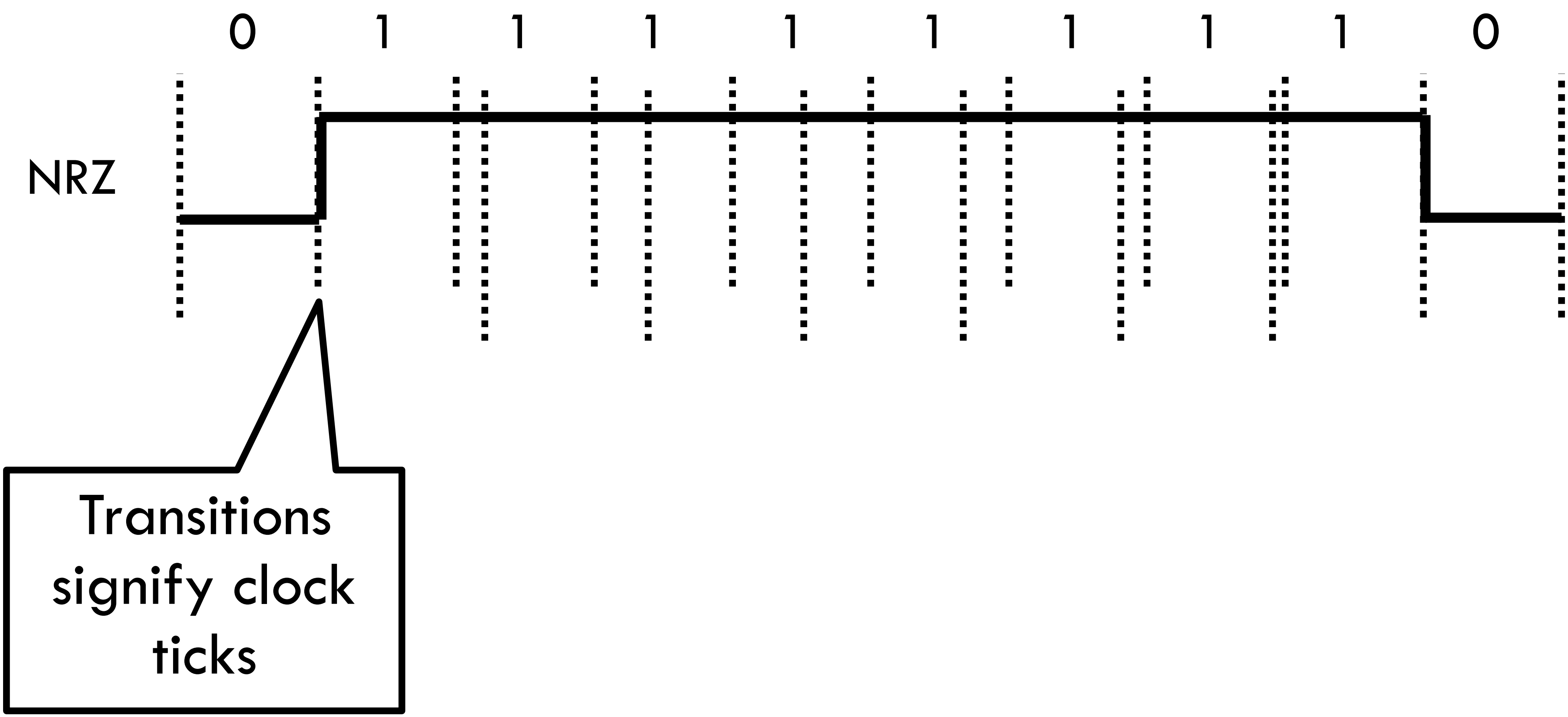
Desynchronization

- Problem: how to recover the clock during sequences of 0's or 1's?



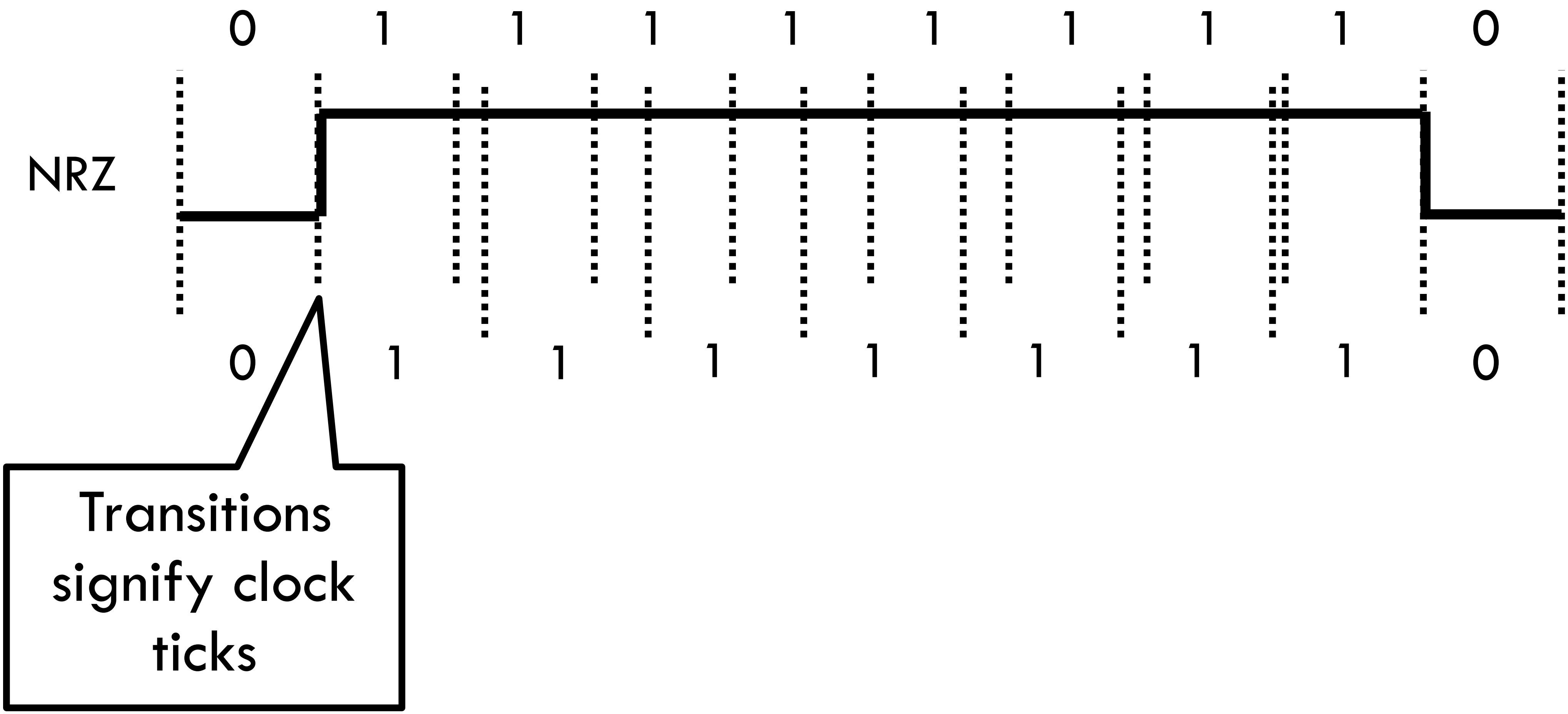
Desynchronization

- Problem: how to recover the clock during sequences of 0's or 1's?



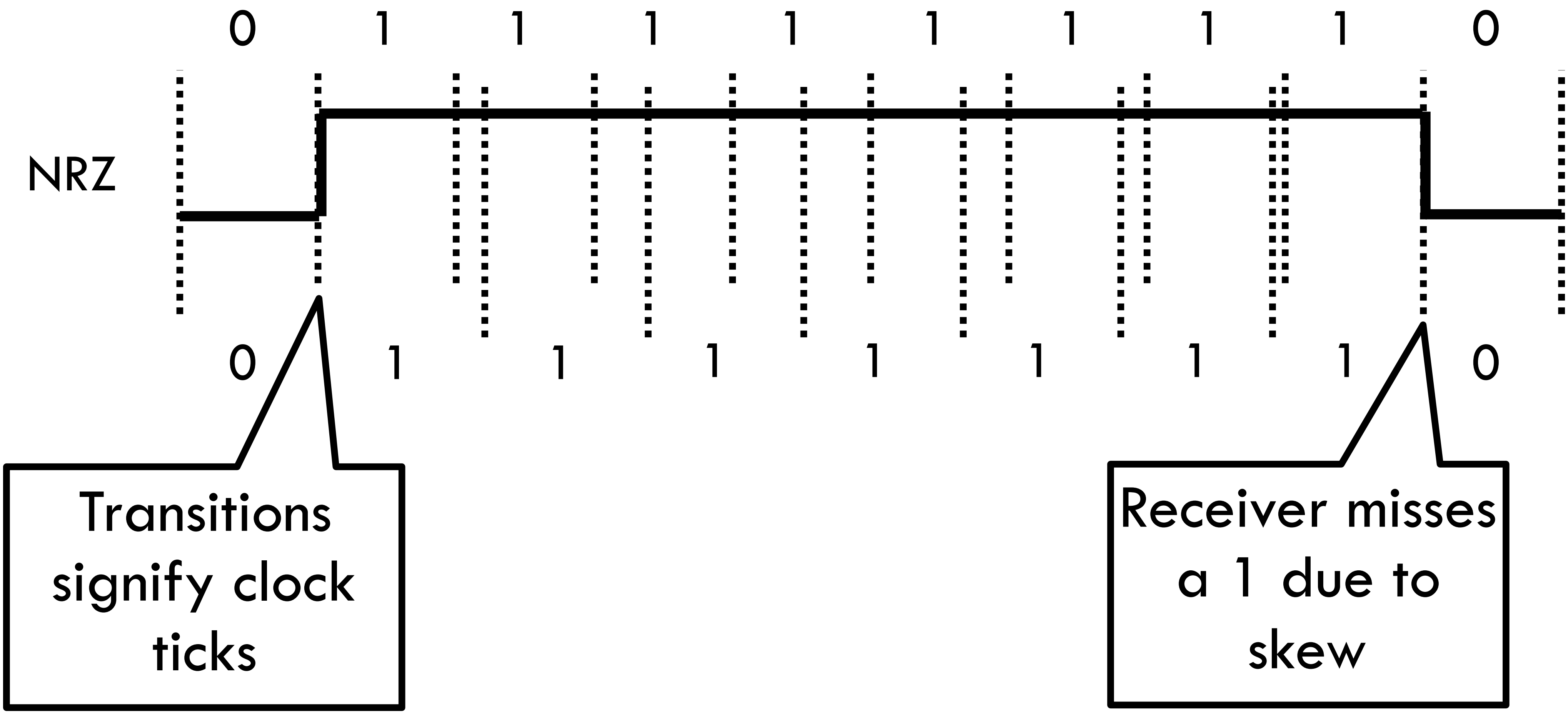
Desynchronization

- Problem: how to recover the clock during sequences of 0's or 1's?



Desynchronization

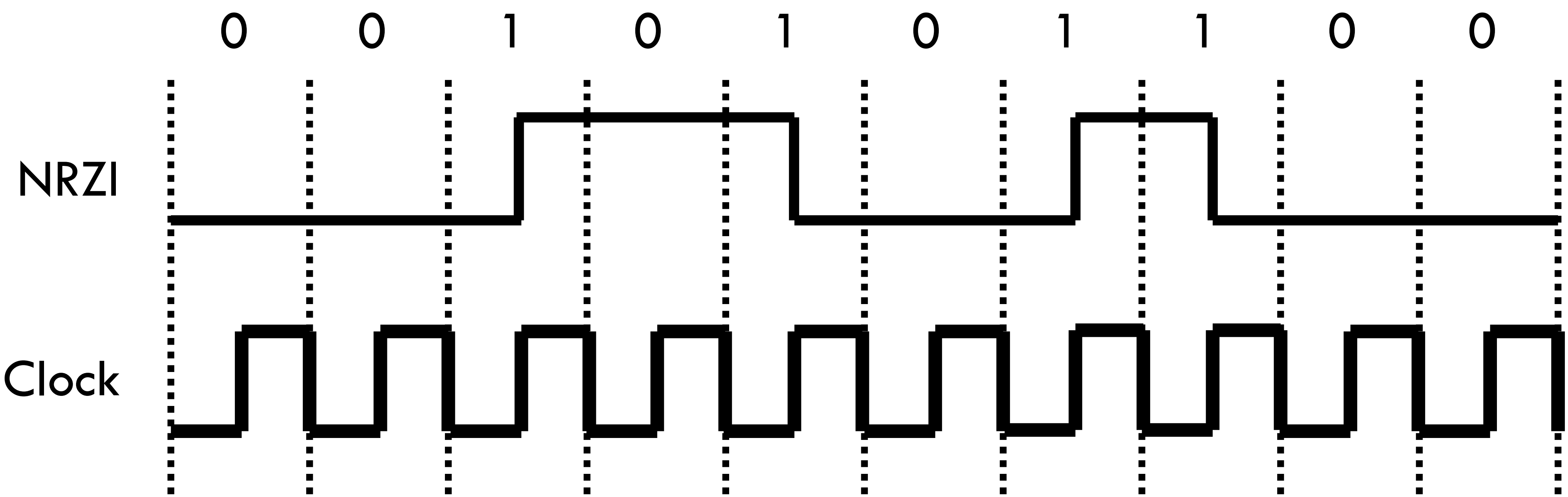
- Problem: how to recover the clock during sequences of 0's or 1's?



Non-Return to Zero Inverted (NRZI)

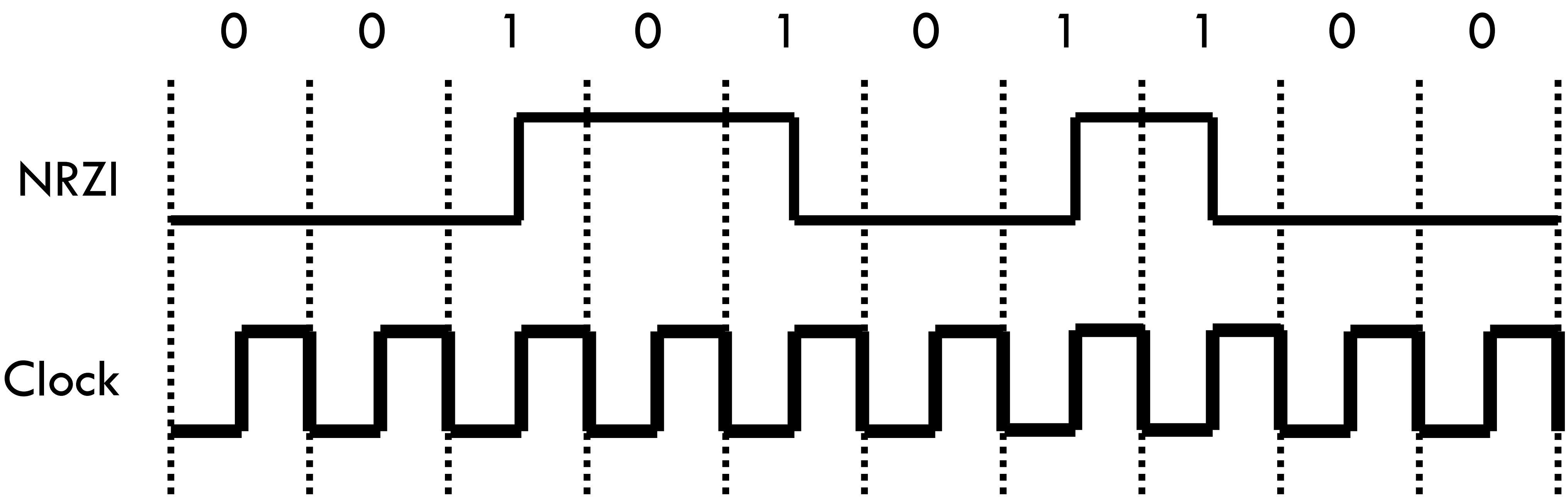
7

- 1 → make transition, 0 → remain the same



Non-Return to Zero Inverted (NRZI)

- 1 → make transition, 0 → remain the same



- Solves the problem for sequences of 1s, but not 0s

4-bit/5-bit (100 Mbps Ethernet)

8

4-bit	5-bit
0000	11110
0001	01001
0010	10100
0011	10101
0100	01010
0101	01011
0110	01110
0111	01111

4-bit	5-bit
1000	10010
1001	10011
1010	10110
1011	10111
1100	11010
1101	11011
1110	11100
1111	11101

4-bit/5-bit (100 Mbps Ethernet)

- Observation: NRZI works as long as no sequences of 0
- Idea: encode all 4-bit sequences as 5-bit sequences with no more than one leading 0 and two trailing 0

4-bit	5-bit	4-bit	5-bit
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

- Tradeoff: efficiency drops to 80%

4-bit/5-bit (100 Mbps Ethernet)

- Observation: NRZI works as long as no sequences of 0
- Idea: encode all 4-bit sequences as 5-bit sequences with no more than one leading 0 and two trailing 0

8-bit / 10-bit used in Gigabit Ethernet

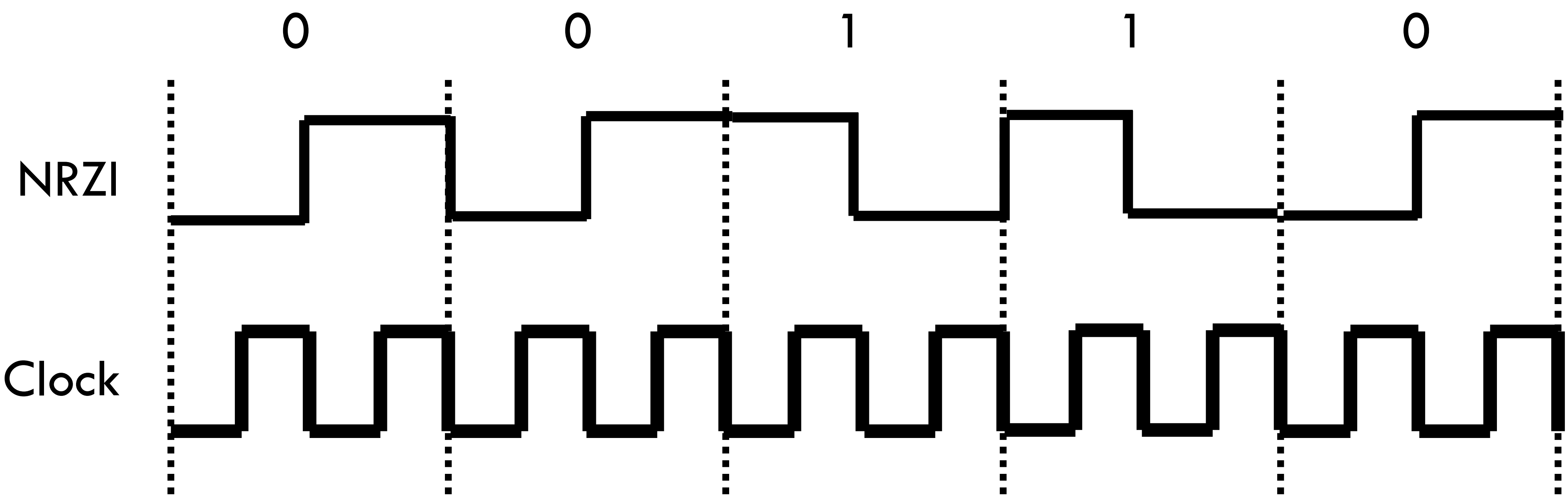
4-bit	5-bit	4-bit	5-bit
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

- Tradeoff: efficiency drops to 80%

Manchester

9

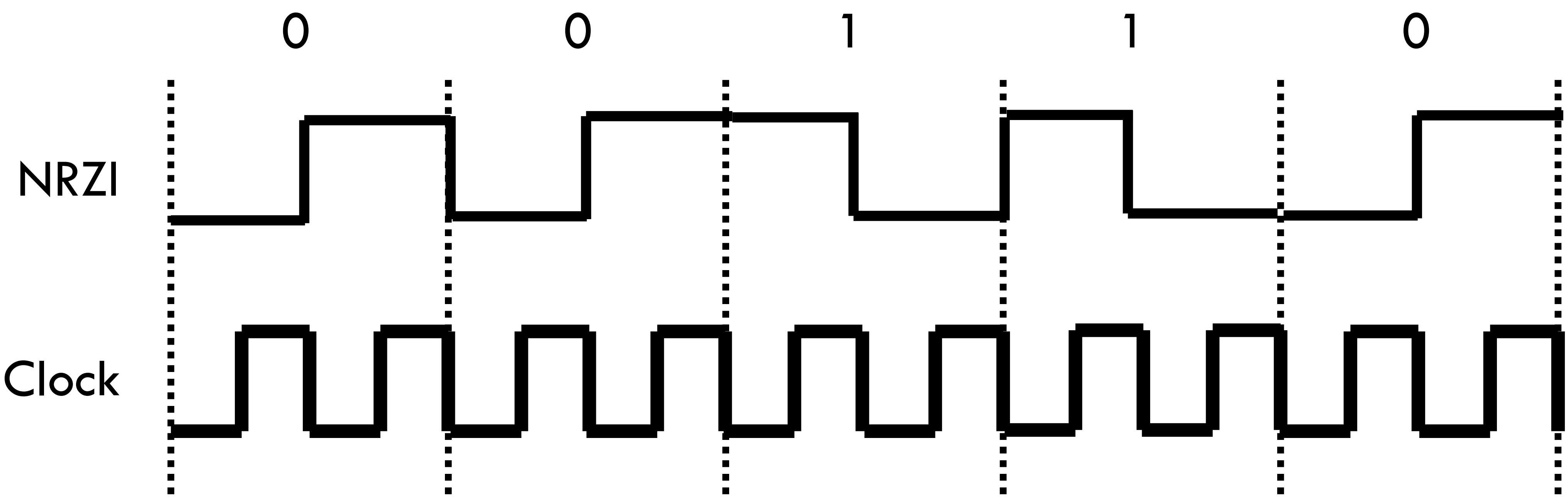
- 1 → high-to-low, 0 → low-to-high



Manchester

9

- 1 → high-to-low, 0 → low-to-high



- Good: Solves clock skew (every bit is a transition)
- Bad: Halves throughput (two clock cycles per bit)

General comment

- Physical layer is the lowest, so…
  - We tend not to worry about where to place functionality
  - There aren't other layers that could interfere
  - We tend to care about it only when things go wrong

- Physical layer characteristics are still fundamentally important to building reliable Internet systems
  - Insulated media vs wireless
  - Packet vs. circuit switched media