「4장: Digital Transmission」

Digital data
→Digital signal

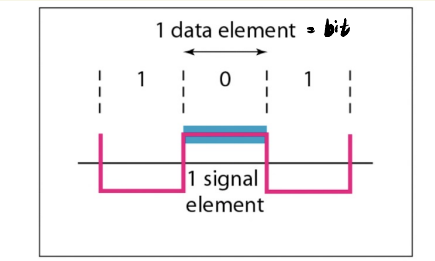
Digital data to Digital signal conversion (Encoder + Decoder)

Techniques

D to D conversion
- Line coding (아날 데이터를 디지털 신호로 바꾸는 기본 방식) ← always needed
- block coding (+α)
- scrambling (+α)

Line coding
개념 정리

Data Element 와 Signal Element



Line coding : Digital data → Digital signal

데이터의 최소 단위 : Data Element = Bit
신호의 최소 단위 : signal Element

data → signal로 변경 시에, 하나의 signal Element가 몇개의 Data Element(bit)를 표현가능한가?
⇒ 이를 Ratio r 이라 한다.

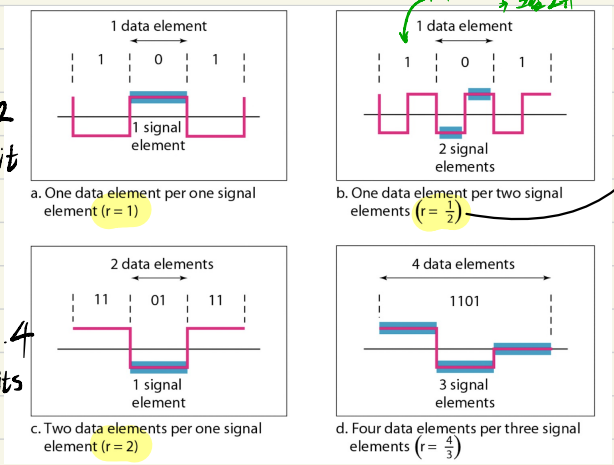
Ratio r

The number of data Elements carried by each signal Element
⇒ 한개의 signal Element가 전달하는 data Elements [bits]의 수

[bits / signal Element] ⇒ 클수록 좋다!

1과 0의 신호 같음
→ 그럼 오류

LV.2
⇒ 1 bit



a. One data element per one signal element (r = 1)

b. One data element per two signal elements $\left(r = \frac{1}{2}\right)$

r이 작은데 왜 써야하지?
→ synchronization (동기화) 때문이다.

LV.4
⇒ 2 bits

c. Two data elements per one signal element (r = 2)

d. Four data elements per three signal elements $\left(r = \frac{4}{3}\right)$

Data rate 와 Signal rate

Data rate == bit rate [bps (bits per second)]

Signal rate == $\frac{\text{band}}{\text{보수}}$ rate [signal Element per second]

signal Element/s * $\frac{\text{Data Element (bits)}}{\text{Signal Element}}$

$$S = C \times \frac{N}{r}, \quad N = \frac{1}{C} S r$$

↳ bit rate [bps]

Line coding의 목표 : data rate 높이고, signal rate 낮추자
⇒ 단위 signal당 보내는 bit 수를 높이자.

Bandwidth

↳ spectrum의 너비

Digital signal은 무한대의 bandwidth가 필요하다.

But! 현재는 infinite한 link가 없다. → finite하지만 effective한 bandwidth 사용

$$B_{min} = C \times \frac{N}{r} \qquad N_{max} = \frac{1}{C} B r$$

지향점

Line coding을 할때 — 지향할 것: Self-synchronization
Built-in-Error Detection
Immunity to Noise
지양할 것: Baseline Wandering
DC Component
Complexity

Self-synchronization

↳ digital signal에 timing 정보 (시작, 중간, 끝)가 들어있다.
⇒ 수신자의 bit 간격과 송신자의 간격이 일치

Built-in-Error Detection

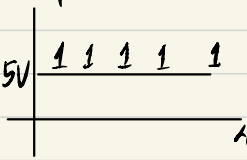
↳ error Detection을 위해, Bit를 추가하여 전송

Baseline Wandering

↳ Baseline : 수신된 신호 '진폭' (파워)의 평균

수신 측에서 전압에 따라 0인지 1인지를 판단하는데,
연속적으로 0 또는 1이 오면 baseline이 흔들려서 0/1 판단을 잘못하는 현상

DC Component



연속된 1을 5V로 계속 보냄
→ f = 0 이 됨.
하드웨어적으로 low frequency를 인식 못하는 경우가 있어서 지양해야.

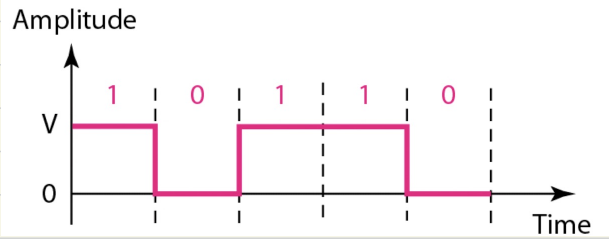
line coding 기법

Line coding
- Unipolar (극성이 1개) → 1이면 전압 줌, 0이면 안줌
- Polar (극성이 2개) → (+)(-) 전압 사용
- Bipolar (+),(-),(0) 사용

Unipolar

NRZ (Non-Return-to-Zero)

↳ 1이면 (+) 전압, 0이면 전압 안줌

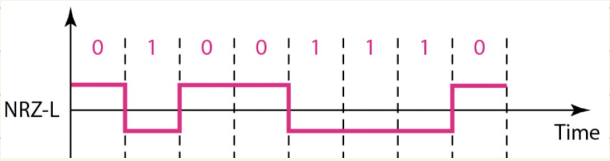


⚠ 단점
- self-synchronization 안됨
- Baseline wandering, DC component 해결 불가
- power 소모 큼

polar

NRZ-L (NRZ-Level)

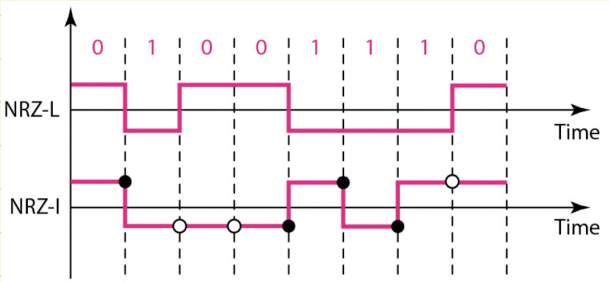
↳ V의 Level이 bit 값 결정 (ex: Level 1 → 1, Level 2 → 0)



⚠ 단점 — Baseline wandering, DC Component 해결못함
self-synchronization 없음

NRZ-I (NRZ-Inversion)

↳ Inversion이 있으면 1, 없으면 0으로 판단한다.



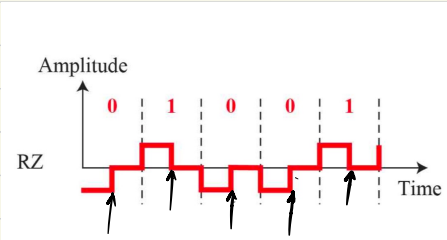
⚠ 단점 — 연속으로 오는 1은 Baseline Wandering, DC Component 해결
But, 연속으로 오는 0은 여전히 해결 불가능

+ self-synchronization 없음

RZ (Return-to-Zero)

↳ 중간에 0으로 되돌아감 (0이던 1이던)

1 : 양(+) → 0 , 0 : 음(-) → 0



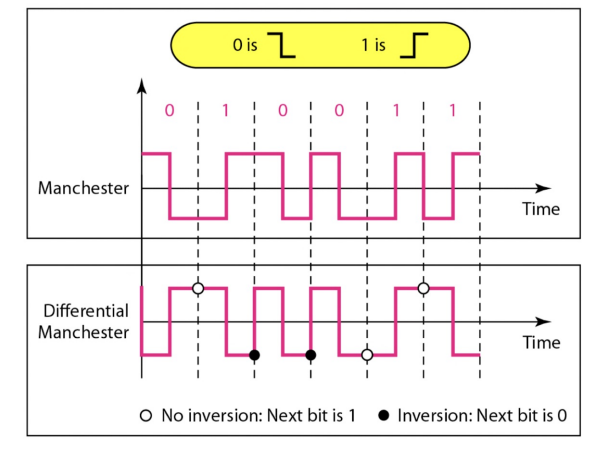
중간 지점을 알 수 있다 ⇒ self-synchronization 가능

+ 연속된 값이 오더라도 Baseline Wandering, DC component 어느정도 해결

⚠ 단점 — Bandwidth, Complexity

Biphase

↳ Manchester와 differential Manchester가 있다.
RZ+NRZ-L RZ+NRZ-I



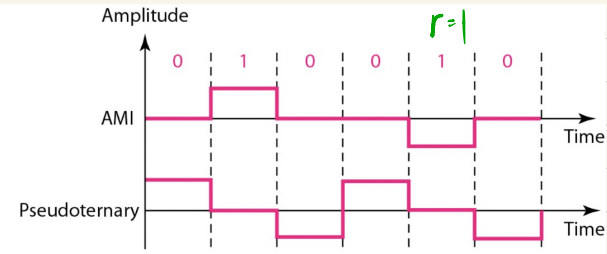
중간 지점을 알 수 있다 ⇒ self-synchronization 가능
+ No Baseline wandering, No DC component

⚠ 단점 — 2 signal Elements가 1개의 bit 표현 ⇒ $r=\frac{1}{2}$ 이다.

Bipolar

AMI & Pseudoternary

↳ 0은 그대로 inversion을 함. 1은 0이다.
↳ 1은 그대로 inversion을 함. 0은 0이다.



No DC Component
Baseline wandering 없음
↳ 0일때는 어차피 0V라

⚠ 단점 — 0 연속일 때 self synchronization 불가

Block coding

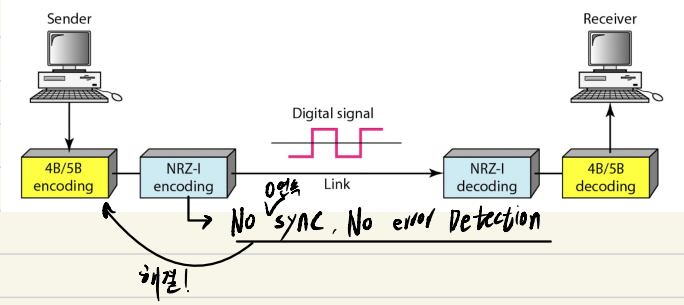
여분의 Bit를 추가하여 self-synchronization, error detecting 개선
Line coding에 +α적인 기법

mBnB

n > m
m개의 bit 원본에 여분을 추가하여 n개로 전송

순서

1. digital data를 m개 bit씩으로 쪼갠다. (division)
2. m-bits group을 n-bits group으로 대체한다 (Substitution)
3. n-bits group을 다 합친다. (combination)



4B/5B

Data Sequence	Encoded Sequence	Control Sequence	Encoded Sequence
0000	11110	Q (Quiet)	00000
0001	01001	I (Idle)	11111
0010	10100	H (Halt)	00100
0011	10101	J (Start delimiter)	11000
0100	01010	K (Start delimiter)	10001
0101	01011	T (End delimiter)	01101
0110	01110	S (Set)	11001
0111	01111	R (Reset)	00111
1000	10010		
1001	10011		
1010	10110		
1011	10111		
1100	11010		
1101	11011		
1110	11100		
1111	11101		

mapping

mapping된 5B에서 연속 0의 최대 개수 : 2

ex) 1110 0001 ⇒ 11100 01001
4회째 3개밖에 연속못함

⇒ self sync 해결!

4B : 2^4개 pattern 5B : 2^5개 pattern
16개 pattern이 mapping이 안되는데? — 절반은 control bit로 mapping
— 나머지가 들어오면 error

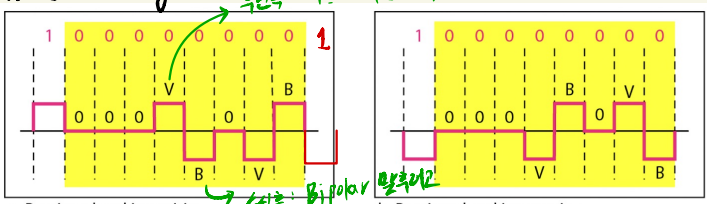
Scrambling

연속된 0을 계속 섞을으로써, 0연속 self-synchro 해결

B8ZS

AMI with scrambling

수신측: 어? 1이면 (-) 나올차례인데 ... 0이구나!



a. Previous level is positive. b. Previous level is negative.

수신측: Bipolar 발견하면 넘어가구나 ... 0이네

연속된 8개의 0은 0000 0000 → 000V B0VB로 바꿈

violation Bipolar

※ 0 7개까진 괜찮다.

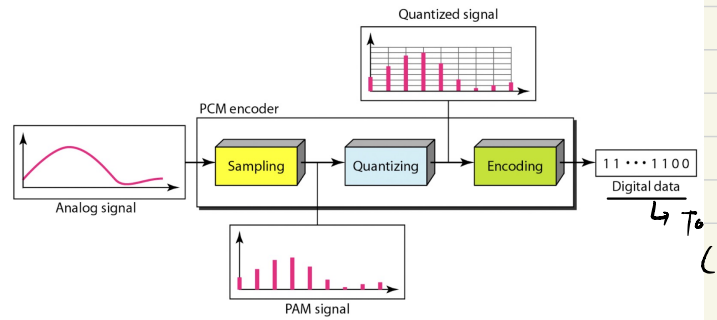
Analog data to Digital signal

☆Codec (coder + decoder) : Analog signal ⟶ digital data

↳ 이 과정에선 어쩔 수 없는 손실이 발생한다.

ATD 기법 중 PCM (Pulse Code Modulation)이 가장 널리 쓰인다.

PCM



↳ To digital signal (Line coding)

PCM은 3과정으로 진행된다.

1. Sampling (PAM: Pulse Amplitude Modulation)
2. Quantizing
3. Encoding

Sampling

Analog signal을 T초마다 한번씩 표본으로 측출한다. ($y=f(x)$에서 $f(x), f(x+T), f(x+2T) \ldots$)

→ == sampling rate [Samples / s]

Then, 얼마나 자주 해야 하는가?

Sampling rate

According to Nyquist theorem

Sampling rate (S) ≥ $2 * f_{high}$

Quantization

Sampling이 진행된 신호를 일말은 digital level로 여차시킨다.

※ Level의 수는 선택 가능하다. (L은 0부터 시작)

if $L = 2^n$ ⇒ sample당 n개의 bit를 표현한다.

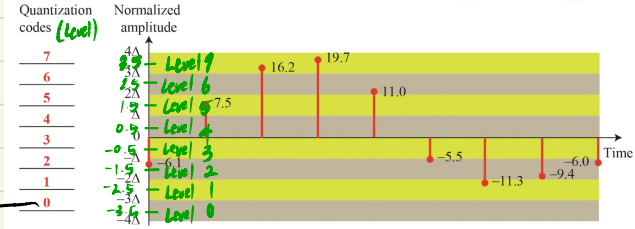
if L이 커지면 ⇒ 정확도 상승, But 데이터 양도 상승

Quantization 순서 ★

하나의 sample에 대하여

1. $(V_{max} - V_{min}) / L$을 구한다. 이 값은 Level간 전압의 차이 Δ가 된다.
2. sample 값 / Δ을 구한다. (Nomalized PAM value)
3. 2의 값을 가장 가까운 Level Value로 양자화 한다. (Nomalized quantized value)
4. 해당 Level이 sample의 Quantization Code이다.

Quantization & encoding of a sampled signal (with V_{min} =-20V, V_{max}=+20V, L=8)



Normalized PAM values	−1.22	1.50	3.24	3.94	2.20	−1.10	−2.26	−1.88	−1.20
Normalized quantized values	−1.50	1.50	3.50	3.50	2.50	−1.50	−2.50	−1.50	−1.50
Normalized error	−0.28	0	+0.26	−0.44	+0.30	−0.40	−0.24	+0.38	−0.30
Quantization code	2	5	7	7	6	2	1	2	2
Encoded words	010	101	111	111	110	010	001	010	010

0부터 시작!

ex) 16.2 sample

$(V_{max} - V_{min}) / L = 5$, ∴ $\Delta = 5$

$16.2/\Delta = 3.24$, ∴ Normalized PAM Value = 3.24

가장 근사한 Level Value : 3.5 , ∴ Normalized quantized value = 3.5

Quantization Code : 7

Encoding

각각의 Quantization Code를 bit로 표현한다.

↳ 몇개의 bit로? $n = \log_2 L$개 bit

Bit rate = Sampling rate * 샘플 당 비트수

[sample/s] * [bits/sample] = [bps]

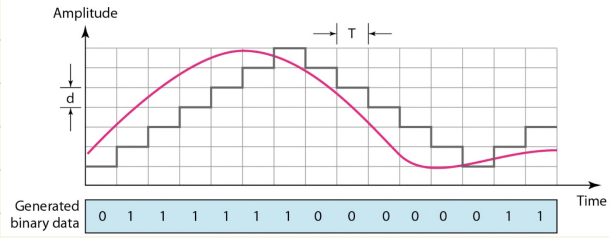
ex) Quantization code

7	→	111
6	→	110
5	→	101
4	→	100
3	→	011
2	→	010
1	→	001
0	→	000

Level이 8개이므로 3개 bit로 표현

Delta modulation

간단하게 Analog signal → digital data 변환



이전보다 Amplitude 증가 ⇒ 1 , 감소 ⇒ 0

데이터의 양이 작다 (Complexity 가 낮다)

⚠ 단점 — A가 확 뛰는 경우 ⎍ 그 증가 폭을 나타낼 수 없음

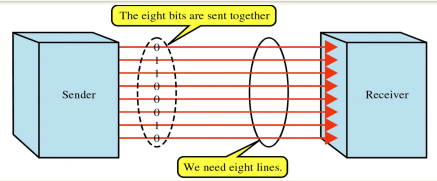
Data Transmission

Digital signal 전송 방식

- Parrarel
- Serial
  - Asynchronous
  - Synchronous
  - Isochronous

동기화 방법에 따라 3종류가 있다.

Parrarel

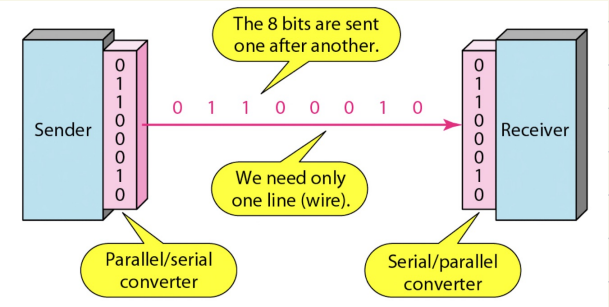


여러개의 bit를 동시에 보낼 (clock purse)

☆ 빠르지만 비싸다 , 단거리에 주로 이용

Serial

each clock purse에 한개의 bit를 내보낼



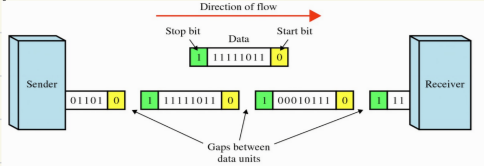
장거리에 사용

Asynchronus Transmission

↳ clock 성능이 좋지 않을 때 사용한다.

시작에 bit 0, 끝에 bit 1을 추가해서 동기화

Byte 단위로 오버헤드를 붙인다



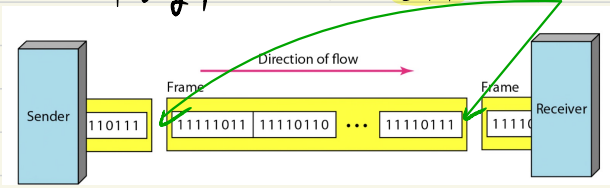
8 bit → 10 bit

정보량 증가!

Synchronous Transmission

↳ clock이 어느정도 잘 맞을 때 사용한다.

시작과 끝에 flag pattern을 넣어서 동기화 ex) 01111110



Frame 단위로 flag를 붙여 보낸다.

01111110

Q! Frame 중간에 Flag pattern이 있다면?

⇒ Bit stuffing / destuffing을 사용한다.

송신측에서 1이 5개 나온 뒤에 0을 붙인다. 01011110

↳ 수신측에서 Flag가 아닌 것을 확인하고 01111110으로 복구

⚠ 10111110 은?

⇒ Bit stuffing은 1이 5개면 무조건 발생. 100111110

↳ 수신에서 알아서 뺄